openATTIC Documentation

Release 1.1.0

it-novum GmbH

September 10, 2015

1 Quick start **3**
1.1 Install . . . 3
1.2 First steps . . . 3

2 Requirements **5**
2.1 Achieving Scalability and High-Availability . . . 5
2.2 Performance . . . 6
2.3 Hardware requirements . . . 6
2.4 System requirements . . . 7
2.5 Required knowledge . . . 7

3 Use Cases **9**
3.1 Fileserver . . . 9
3.2 Virtualization . . . 10
3.3 Cloud Storage . . . 11
3.4 Object Storage . . . 12
3.5 Storage consolidation . . . 12
3.6 Backup . . . 13

4 Installation and Upgrade Guides **15**
4.1 Preparing the Installation . . . 15
4.2 Demo VMs . . . 17
4.3 Step by step installation guides . . . 29
4.4 Package-based installation on Debian and Ubuntu . . . 50
4.5 Installing an openATTIC cluster . . . 51
4.6 Joining openATTIC to a Windows domain . . . 54
4.7 Configuring Authentication and Single Sign-On . . . 55
4.8 Additional openATTIC Modules . . . 57
4.9 Maintenance . . . 61

5 Implementing the Use Cases **63**
5.1 Implementing a file server . . . 63
5.2 Implementing a virtualization storage . . . 66
5.3 Implementing Cloud Storage . . . 69

6 User Manual **71**
6.1 Status . . . 71
6.2 Storage . . . 77
6.3 LUNs (SAN) . . . 89
6.4 Shares . . . 95
6.5 Services . . . 99

6.6 System . . . 102
6.7 Personal Settings . . . 108
6.8 Shutdown . . . 110
6.9 Menu shortcut bar . . . 111
6.10 Hiding the menu tree . . . 112

7 Integration **115**
7.1 Cloud Connectors . . . 115
7.2 XML-RPC API . . . 115
7.3 Integration Tutorial . . . 119

8 Developer documentation **129**
8.1 Setting up a development system . . . 129
8.2 RPC API . . . 130
8.3 openATTIC Core . . . 131
8.4 System API . . . 131
8.5 Integration Testing . . . 133
8.6 Submitting code to openATTIC . . . 133

9 Indices and tables **135**

The times when storage was considered a server-based resource and every system needed to have its own hard drives are long gone. In modern data centers central storage systems have become ubiquitous for obvious reasons. Centrally managed storage increases flexibility and reduces the cost for unused storage reserves. With the introduction of a cluster or virtualization solution shared storage becomes a necessity.

This mission-critical part of IT used to be dominated by proprietary offerings. Even though mature open source projects may now meet practically every requirement of a modern storage system, managing and using these tools is often quite complex and is mostly done decentrally.

openATTIC is a full-fledged central storage management system. Hardware resources can be managed, logical storage areas can be shared and distributed and data can be stored more efficiently and less expensively than ever before – and you can control everything from a central management interface. It is no longer necessary to be intimately familiar with the inner workings of the individual storage tools. Any task can be carried out using the intuitive openATTIC interface.

CHAPTER

ONE

QUICK START

OK, so you just want to get going without all the gory details? Then here you go.

1.1 Install

Installation for Debian Wheezy:

```
apt-key adv --recv --keyserver hkp://keyserver.ubuntu.com A7D3EAFA
echo deb http://apt.open-attic.org/ wheezy main > /etc/apt/sources.list.d/openattic.list
apt-get update
apt-get install openattic
oaconfig install
oaconfig add-disk /dev/sdX vgsomething
```

And like this for Ubuntu Trusty (14.04):

```
apt-key adv --recv --keyserver hkp://keyserver.ubuntu.com A7D3EAFA
echo deb http://apt.open-attic.org/ trusty main > /etc/apt/sources.list.d/openattic.list
apt-get update
apt-get install linux-image-extra-`uname -r` openattic
oaconfig install
oaconfig add-disk /dev/sdX vgsomething
```

That's it, now go to `http://yourbox/openattic` and have fun!

See also:

Installation and Upgrade Guides if you need help with preparing the installation, the installation itself or if you need to know how to setup an openATTIC cluster.

1.2 First steps

1. Take a look at the "Volume pool management" panel. It should display the volume group you just added with the `oaconfig add-disk` command.
2. On the "Volume management" panel, you can create volumes. Create one or two, trying out different file systems.
3. Check out the modules for *CIFS* and *NFS* shares. Create some shares and see if you can access them.

CHAPTER

TWO

REQUIREMENTS

When planning and setting up a storage system, there are lots of things to be considered beforehand: How much data do you need to store? How much growth is to be expected? What performance requirements do you have? Which kind of administrative tasks will your staff be able to handle?

The following chapters will give you an overview of the parameters and limitations you should be aware of, considering both overall system design questions as well as hardware and software requirements.

2.1 Achieving Scalability and High-Availability

In the past several years the amount of data to handle has grown markedly, with the result that your storage system will also have to grow to meet the increased demand. For starters, you can add new disk shelves to the existing system. But one single server will hit a limit someday, and you'll soon find yourself in need of better failure tolerance. Scaling up a single system is therefore not a long-term viable solution.

openATTIC supports scaling out by combining a set of machines to a cluster. There are multiple options for this:

2.1.1 Simple cluster

Installing multiple nodes and combining them into a simple cluster will allow you to manage a multitude of storage systems as easily as a single large one. The openATTIC GUI does not care whether the devices you are configuring are local to the machine you are logged in to, or remote on other storage nodes in the same cluster: The configuration will just work transparently.

However, your clients will still need to connect to the correct storage node in order to access data, and volumes will be limited in size to the amount of storage on a single node. Furthermore, this setup does not tolerate the failure of a storage host, only the failure of individual disks is accounted for.

2.1.2 Mirrored cluster

If the limitations of a simple cluster setup are unacceptable, you can go one step further and install a mirrored cluster. This setup provides failure tolerance by mirroring all data of one node to a standby node which gets activated in case the primary node fails. The clients will not be aware of the failover process.

However, this setup cannot be easily modified or extended and will only allow you to use the space provided by one node at a time.

2.1.3 Advanced cluster

If you want to be able to modify the cluster structure more easily by simply adding or removing nodes, you can achieve that using the advanced cluster setup. This configuration combines the storage space of all nodes in the cluster and automatically distributes copies of data across different nodes, providing both scalability and failure tolerance in the process.

To the clients, the storage cluster looks like a single entity, providing petabytes of storage space at the click of a button. This way, you're not limited to the space of one storage node in any way.

2.2 Performance

When you're centralizing storage for a multitude of machines, you have to make sure the storage system is able to take the load. This cannot be achieved simply by buying more and better hardware: Scaling up vertically by upgrading the hardware of a single machine gets pretty expensive beyond a certain point. Scaling out horizontally by buying more machines is fair enough regarding the price, but at the end of the day, it always comes down to a single user waiting for a single disk to do stuff, so this approach does not do anything to increase that particular user's experienced performance.

The only really effective way to keep performance high is to avoid making the mistakes which degrade it. openATTIC has proven its worth in real-world datacenters, running as central storage for a virtualization cluster where latency is critical.

Throughout this guide, you will find hints and warnings about performance issues. Taking these into account is strongly advisable to make the best of your hardware.

2.3 Hardware requirements

openATTIC is designed to run on commodity hardware, so you are not in any way bound to a specific vendor or hardware model. However, there are a couple of things you should be aware of when designing the system.

1. Buy an enclosure with enough room for disks. The absolute minimum recommendation is twelve disks, but if you can, you should add two hot-spares, so make that fourteen. For larger setups, use 24 disks.

   Warning: Any other number of disks will hinder performance.

2. Are you building a storage backend for virtualization? If so, you will require SAS disks, a very clean setup and a good caching mechanism to achieve good performance.

   Note: Using SSDs instead of SAS disks does not necessarily boost performance. A clean setup on SAS disks delivers the same performance as SSDs, and an unclean SSD setup may even be slower.

3. If the enclosure has any room for hot spare disks, you should have some available. This way a disk failure can be dealt with immediately, instead of having to wait until the disk has been replaced.

   Note: A degraded RAID only delivers limited performance. Taking measures to minimize the time until it can resume normal operations is therefore highly advisable.

4. You should have some kind of hardware device for caching. If you're using a RAID controller, make sure it has a BBU installed so you can make use of the integrated cache. For ZFS setups, consider adding two SSDs.

   Note: When using SSDs for caching, the total size of the cache should be one tenth the size of the device being

cached, and the cache needs to be ten times faster. So,

- only add a cache if you *have to* – no guessing allowed, measure!
- don't make it too large
- don't add an SSD cache to a volume that is itself on SSDs

---

5. Do you plan on using replication in order to provide failure tolerance? If so, ...
   - you will require the *same* hardware for *all* of your nodes, because when using synchronous replication, the slowest node limits the performance of the whole system.
   - make sure the network between the nodes has a low latency and enough bandwidth to support not only the bandwidth your application needs, but also has some extra for bursts and recovery traffic.

     Note: When running VMs, a Gigabit link will get you pretty far. Money for a 10GE card would be better spent on faster disks.

   - you should have a dedicated line available for replication and cluster communication. There should be *no other active components* on that line, so that when the line goes down, the cluster can safely assume its peer to be dead.
6. Up to the supported maximum of 128GB per node, add as much RAM as you can (afford). The operating system will require about 1GB for itself, everything else is then used for things like caching and the ZFS deduplication table. Adding more RAM will generally speed things up and is always a good idea.

2.4 System requirements

1. openATTIC is designed to run on Linux.
2. Supported Linux distributions are:
   - Debian Wheezy
   - Ubuntu Precise (12.04 LTS)
   - Ubuntu Trusty (14.04 LTS)
   - Univention Corporate Server 3.1
3. In order to use ZFS, make sure ZFS on Linux is available for your distribution. (For the distributions listed above, it is.)
4. FibreChannel support requires at least Linux 3.5.

2.5 Required knowledge

openATTIC is targeted towards administrators. If you're looking for an end-user interface, you should take a look at *Cloud Storage*.

Since storage is always configured as a part of some larger system, setting it up requires some knowledge about storage in general and the system you're going to use it for. Knowing what your target system's requirements are is therefore necessary in order to build a system that delivers the performance and capacity you need. Check out the *Use Cases* section for more information.

You should also have a basic understanding of the difference between file-based and block-based storage protocols and their applications.

You do not require a deep understanding of how the software components that openATTIC uses are configured and how they interact.

CHAPTER

THREE

USE CASES

Being one of the key components of your infrastructure, openATTIC can be used in a number of ways ranging from simple shared file servers up to high-performance virtualization clusters. This chapter gives you an overview of possible configurations, other products openATTIC can be combined with, and the benefits and limitations of the various setup variations.

3.1 Fileserver

Central file storage has been a requirement in the business world from the day networks were invented. And it makes sense, too: Having files available on the network strengthens both collaboration and independence, because you can access your files from anywhere without having to ask anyone for a copy.

When using openATTIC for this kind of central storage, it has to fulfill a set of requirements.

1. Access to files needs to be authenticated in a secure manner.
2. Users need to be able to manage file permissions themselves in a well-defined manner.
3. User management needs to be synchronized with the rest of the infrastructure.
4. Single sign-on should work, so people don't need to enter their passwords all the time.

These basic requirements can easily be met using the CIFS protocol in a Windows domain. The CIFS protocol has been specifically designed for this use case and provides strong authentication and authorization mechanisms. Integrating openATTIC in a Windows domain is easy, and the domain then provides centralized user management combined with Single Sign-On.

Setting up a Windows domain used to require a Windows Server license and was therefore only worthwhile for businesses. But since the release of Samba 4, all you need is a Linux box and you can get a Windows domain running within minutes.

Of course, openATTIC can be set up in multiple ways that not only provide the necessities, but allow for some extra features to be added on top.

See also:

Implementing a file server.

3.1.1 Quick and dirty

The simplest setup would be using an LV formatted with the Ext4 file system. This is a very affordable and quick way to get started. However, scalability is limited and you don't have any failure tolerance outside of what you can provide within a single server through technologies such as RAID.

3.1.2 Clustered setup

Of course, a file server will also benefit from clustering as outlined in the *Achieving Scalability and High-Availability* section.

3.1.3 Snapshots

Accidental deletion of files is something that happens every day, especially in a shared environment where many people have access to the file system. openATTIC provides a snapshot mechanism that automatically exports snapshots of the file system as a hidden subdirectory, so that users can easily recover deleted files from snapshots. Of course, snapshots enforce the same permissions as the file system itself, so users can only restore snapshots if they were able to access the original file as well.

3.1.4 Mirror servers

While your infrastructure continues to grow, you will get to the point where you have a bunch of systems that need to download the same files over and over when installing software or applying updates. Also, your collection of ISO images and virtual machine templates will continue to grow, especially if you're using Cloud systems like OpenStack, which allow the users to create their own images.

Software repositories usually use the HTTP protocol. openATTIC fully supports exporting volumes via HTTP, allowing you to set up a highly available software repository that gives you the benefit of maximum bandwidth.

3.2 Virtualization

Virtualization systems enforce stringent requirements upon their storage systems because if storage breaks, lots of other components are affected and half of your infrastructure is down. openATTIC has proven its worth in real-life datacenters running all kinds of setups, from single-node setups with only a handful of virtual machines up to multi-datacenter setups with thousands of VMs.

The most subtle requirement is low latency. A slow storage system will affect each and every server that uses it, causing performance issues everywhere that cannot be easily traced to one single source because they are ubiquitous. This means that when setting up the system, care needs to be taken in order to meet the performance requirements by making sure no hardware resources go to waste.

Eliminating single points of failure is another important goal: You need to be able to tolerate failure of any single component without affecting the availability of the whole system. And if done right, you also won't have to take anything offline while adding or even removing storage nodes. For more information, see the *Achieving Scalability and High-Availability* chapter.

Snapshot technologies can be leveraged in order to speed up backup and restore processes, and even to run backup processes on a snapshot instead of the original machine. See the *Target Offloading* chapter for details.

See also:

Implementing a virtualization storage.

3.2.1 oVirt

openATTIC has been extensively tested and used in conjunction with the oVirt virtualization platform, a recently published case study can be found here. oVirt is a fully open source alternative to enterprise virtualization systems such as VMware vSphere, based on the KVM virtualization technology.

3.2.2 VMware vSphere

vSphere is VMware's virtualization operating system, which is widely used in data centers today. openATTIC fully supports providing storage for VMware.

3.3 Cloud Storage

As stated in *Required knowledge*, openATTIC is targeted towards administrators. In order to provide products usable by end-users, a couple of products have been developed recently that hide all the complexity. They achieve this by only asking the user which size the volume needs to be and *completely* automating the creation and management processes. openATTIC has been designed from the start to be the perfect storage backend for these systems.

Cloud storage comes in multiple flavors. For one, you can register for a service that stores files for you online, keeping them synchronized on multiple devices at home or at work, and allowing you to easily share them with other people. But Cloud storage is also used with virtualization, allowing you to add storage to a virtual server easily whenever you need it.

See also:

Implementing Cloud Storage.

3.3.1 IaaS: OpenStack, openQRM

The term "Infrastructure as a Service" refers to systems that allow untrained end users to create virtual servers and manage their own infrastructure as they need it, without requiring administrator privileges or manual work to be done. IaaS products take care of the configuration and at the same time ensure that the policies defined by the administrators are enforced.

Examples of such systems are:

- OpenStack is an IaaS project that aims to provide a "ubiquitous open source cloud computing platform for public and private clouds." It consists of a series of products, one of which is a storage configuration system named Cinder, which creates volumes as per the user's request and makes them available for use on virtual servers.

  Cinder can be configured to use openATTIC as its storage backend through a special driver that is developed and actively supported by the openATTIC team.

- openQRM provides a standardized server deployment workflow, integrating and combining both common and custom system administration tools and solutions into a powerful single management console for your complete IT-service Infrastructure. openATTIC integrates into openQRM via a plugin that is developed and actively supported by the openATTIC team.

- Another infrastructure project that is gaining traction is Ganeti, which can also be configured to use openATTIC through its ExtStorage interface.

3.3.2 Automation

The key component of Cloud storage is automation. openATTIC has been designed with Automation in mind from the very beginning and provides an XML-RPC interface, through which *every* feature that openATTIC provides can be configured remotely. Using openATTIC as the foundation for automated deployment will give you maximum flexibility. Please refer to the *Integration Tutorial* for a demonstration of how this works.

3.3.3 Synchronized file storage: OwnCloud

openATTIC can be used as a storage backend for OwnCloud to provide a shared file storage and synchronization service. Your users will get the benefit of having the latest version of their files on all devices and being able to easily share them with others. Administrators won't need to worry about where the storage comes from, because OwnCloud seamlessly joins multiple volumes together and takes care of the decision on where to put files.

3.4 Object Storage

Object storage systems provide a way of storing data as a large array of objects. The object storage system does not care about the structure of these objects; structure is left to whatever system is using the object store. This can be an emulation of a classic file system, but it does not have to be that way; any structure the application requires is possible because the object store just doesn't care.

Object stores focus on scalability, high-availability and redundancy and leave the rest to the application.

Note: We are currently working on this part of our documentation. An update will be available soon.

3.4.1 Swift

OpenStack Object Storage (Swift) is a scalable redundant storage system. Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster.

3.4.2 Ceph

Ceph does the same thing, but also provides block storage and file-system based storage in one fell swoop and allows for more sophisticated control over the data placement and redundancy.

3.4.3 Hadoop

Built on top of HDFS, Hadoop allows to apply large-scale algorithms to the data stored in an object store and to generate reports.

3.5 Storage consolidation

When running a datacenter for quite some time, storage boxen accumulate and stuff starts getting confusing. openATTIC allows to consolidate storage from all kinds of systems, openATTIC or otherwise, to be visible and manageable by a single openATTIC system.

Note: We are currently working on this part of our documentation. An update will be available soon.

3.6 Backup

Backup is a standard procedure in every datacenter that protects against data loss and corruption events and allows data to be restored. Data corruption or loss does not only occur through hardware failures or software errors: By far the most data is lost as a result of human errors like accidental deletion. While *Snapshots* are a useful short-term way to handle that situation, they require the data loss to be noticed within a short period of time. Backups are usually kept longer, so the time to notice data loss is extended.

The downside is that lots of storage space is required, which is why magnetic tapes are still the predominant storage medium for long-term backups. Tapes are however more of a hassle and take longer to restore data from than snapshots, which is why they alone are not a satisfactory solution.

Using disks for a short-term and tapes for a long-term backup is the ideal solution. Not only are snapshots a much nicer way of restoring data, through deduplication, the storage system can actually reduce the amount of data that needs to be stored. And since it acts like a cache device towards the tape library, the life of the tapes is increased because technical problems like shoe-shining can be avoided.

Integrating openATTIC into a backup solution is easy because it acts like a virtual tape library, using the same protocol that the backup software uses to communicate with the actual tape library – the backup software doesn't even notice the difference, while you get to enjoy all the benefits.

3.6.1 SEP sesam

Once you have lots of tapes and machines you're backing up data from, you're going to require a flexible management solution with a searchable index in order to keep the amount of data manageable. SEP sesam is an enterprise-ready backup solution that supports backing up and restoring operating systems, hypervisors, applications, databases and data.

3.6.2 Target Offloading

Apart from easy recovery, snapshots can also be used to take the load of the backup process off of the target system. To achieve that, a snapshot is taken from the live system and made available on a standby system, which is then targeted by the backup software. That way you don't need to freeze the live system during the backup period and there's no additional load the live system has to take: It can just continue serving your customers uninhibited.

3.6.3 Consistent Application Backups

In order to create useful snapshots, it is important to take into account the fact that data is stored in different places at any point in time. Virtual Machines and databases have a significant amount of state data in RAM, and without synchronizing this data to disk before creating the snapshot, the snapshot will not be consistent and therefore useless.

If you want the snapshot to be consistent, the application will need to be informed, so it can synchronize its data prior to the snapshot. openATTIC has a plugin mechanism that allows *SnapApps* to be plugged into the system that handle the communication with applications.

MSSQL

MSSQL databases support a snapshot mechanism based on VSS. The openATTIC SnapApp leverages this mechanism in order to create a consistent snapshot of the database.

VMware

VMware vSphere supports snapshotting virtual machines, including file system synchronization and storing RAM data. The virtual machine is frozen, stored to disk, and then unfrozen. The openATTIC SnapApp for VMware orchestrates this mechanism in order to create a consistent snapshot of virtual machines.

CHAPTER

FOUR

INSTALLATION AND UPGRADE GUIDES

This section guides you through the necessary system preparation and the installation process of the openATTIC software. Advanced installation steps like joining a Windows Domain or adapting the selection of installed modules are covered as well.

4.1 Preparing the Installation

Before installing openATTIC, there are a couple of things you should be aware of when planning the system.

4.1.1 Hardware / Physical setup

1. Always dedicate two disks to a RAID1 for the system. It doesn't matter if you use hardware or software RAID for this volume, just that you split it off from the rest.

   Note: You can also use other devices to boot from if they fit your redundancy needs.

2. When using hardware RAID:

   (a) Group the other disks into RAID5 arrays of exactly 5 disks each with a chunk size (strip size) of 256KiB. Do *not* create a partition table on these devices. If your RAID controller does not support 256KiB chunks, use the largest supported chunk size.

   (b) Using `mdadm`, create a Software-RAID0 device on exactly two or four of your hardware RAID devices. Again, do *not* create a partition table on the resulting MD device. Make sure the chunk size of the RAID0 array matches that of the underlying RAID5 arrays.

   Note: This way, you will not be able to add more than 20 disks to one PV. This is *intentional*. If you need to add more disks, create multiple PVs in the same manner.

   (c) Using `pvcreate`, create an LVM Physical Volume on the MD device and add it to a VG using `vgcreate` or `vgextend`.

   (d) Do *not* mix PVs of different speeds in one single VG.

3. When using ZFS:

   You will need to specify the *complete* layout in the `zpool create` command, so before running it, consider *all* the following points.

   (a) Group exactly six disks in each raidz2. Use multiple raidz2 vdevs in order to add all disks to the zpool.

   (b) When adding SSDs, add them as *mirrored log* devices.

(c) Set the mount point to `/media/<poolname>` instead of just `/<poolname>`.

(d) Do *not* use `/dev/sdc` etc, but use `/dev/disk/by-id/...` paths instead.

So, the command you're going to use will look something like this:

```
zpool create -m /media/tank tank \
      raidz2 /dev/disk/by-id/scsi-3500000e1{1,2,3,4,5,6} \
      raidz2 /dev/disk/by-id/scsi-350000392{1,2,3,4,5,6} \
      log mirror /dev/disk/by-id/scsi-SATA_INTEL_SSD{1,2}
```

4.1.2 Operating System

1. Disable swap.
2. Make sure the output of `hostname --fqdn` is something that makes sense, e.g. `srvopenattic01.example.com` instead of `localhost.localdomain`. If this doesn't fit, edit `/etc/hostname` and `/etc/hosts` to contain the correct names.
3. In a two-node cluster, add a variable named `$PEER` to your environment that contains the hostname (*not* the FQDN) of the cluster peer node. This simplifies every command that has something to do with the peer. Exchange SSH keys.
4. In pacemaker-based clusters, define the following aliases to make life easier:

   ```
   alias maint="crm configure property maintenance-mode=true"
   alias unmaint="crm configure property maintenance-mode=false"
   ```

5. After setting up MD raids, make sure `mdadm.conf` is up to date. This can be ensured by running these commands:

   ```
   /usr/share/mdadm/mkconf > /etc/mdadm/mdadm.conf
   update-initramfs -k all -u
   ```

6. Install and configure an NTP daemon.
7. You may want to install the `ladvd` package, which will ensure that your switches correctly identify your system using LLDP.
8. Make sure `/etc/drbd.d/global_common.conf` contains the following variables:

   ```
   disk {
       no-disk-barrier;
       no-disk-flushes;
       no-md-flushes;
   }

   net {
       max-buffers 8000;
       max-epoch-size 8000;
   }

   syncer {
       al-extents 3389;
   }
   ```

4.1.3 Scheduler

The disk scheduler's job is to reorder IO requests before they're sent to the storage system, which has a dramatic impact on performance. Done right, it will reduce write latency; done wrong, it will wreak havoc. So, the scheduler needs to be configured correctly.

1. If you're using Hardware RAID, make sure the default scheduler is set to `deadline`. This can be verified using the following command:

   ```
   $ grep . /sys/class/block/sd?/queue/scheduler
   /sys/class/block/sda/queue/scheduler:noop [deadline] cfq
   /sys/class/block/sdb/queue/scheduler:noop [deadline] cfq
   /sys/class/block/sdc/queue/scheduler:noop [deadline] cfq
   ```

   If the CFQ scheduler is selected instead, edit `/etc/default/grub`, find the line that defines the `GRUB_CMDLINE_LINUX_DEFAULT` variable, and make sure it contains the `elevator=deadline` option. For example:

   ```
   GRUB_CMDLINE_LINUX_DEFAULT="quiet elevator=deadline"
   ```

   Then run `update-grub` and reboot.

2. If you have SSDs, make sure their scheduler is set to `noop`.
3. For everything else, use `cfq`.

Switching schedulers for different devices can be achieved using a script like the following, for instance as a part of `/etc/rc.local`:

```
# Set twraid schedulers to deadline
for disk in /dev/disk/by-id/scsi-3600050e?????????????????????????; do
   sdx=$(basename `readlink $disk`)
   echo deadline > /sys/class/block/$sdx/queue/scheduler
done

# Set SSD schedulers to noop
for disk in /dev/disk/by-id/scsi-SATA_INTEL_SSD?????????????????????; do
   sdx=$(basename `readlink $disk`)
   echo noop > /sys/class/block/$sdx/queue/scheduler
done

# Set 15k SAS disks' schedulers to cfq
for disk in /dev/disk/by-id/{scsi-3500000e1????????,scsi-35000039?????????}; do
   sdx=$(basename `readlink $disk`)
   echo cfq > /sys/class/block/$sdx/queue/scheduler
done
```

Note: When writing such a script, ensure that the wildcards only match the actual disks, and not any partitions *on* them. So, just using `/dev/disk/by-id/something*` will usually not suffice.

4.2 Demo VMs

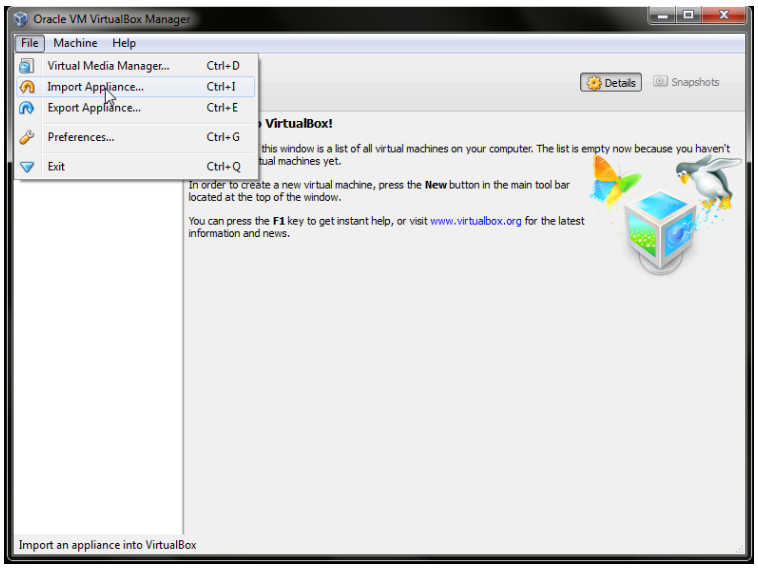
Integrate an openATTIC demo for virtualbox or vmware.

See also

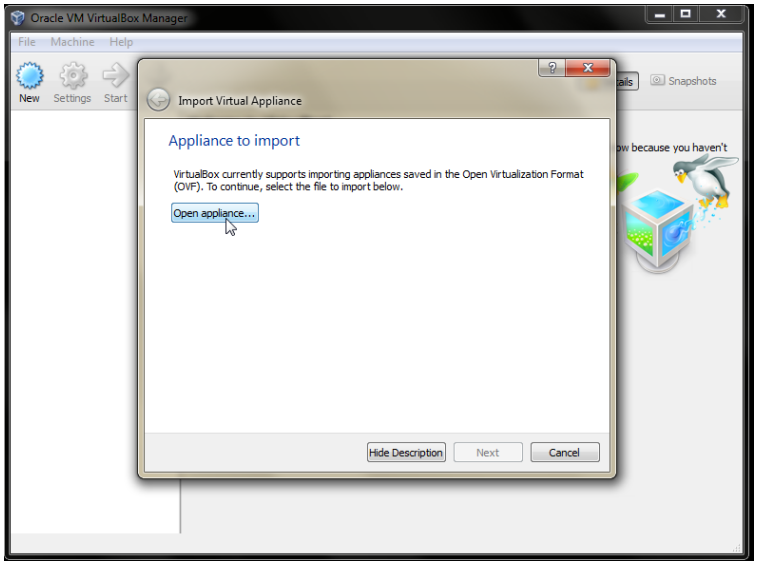
4.2.1 Demo VM for VirtualBox

The openATTIC demo vm version 1.1 or 1.2 can be downloaded from http://apt.open-attic.org/vm/ or from http://www.open-attic.org/downloads and integrated into VirtualBox Manager as follows:

1. open VirtualBox and click “File” –> “Import Appliance”


Oracle VM VirtualBox Manager
File
Machine
Help
Virtual Media Manager... Ctrl+D
Import Appliance... Ctrl+I
Export Appliance... Ctrl+E
Preferences... Ctrl+G
Exit Ctrl+Q
Details
Snapshots
VirtualBox!
this window is a list of all virtual machines on your computer. The list is empty now because you haven't
tual machines yet.
In order to create a new virtual machine, press the New button in the main tool bar
located at the top of the window.
You can press the F1 key to get instant help, or visit www.virtualbox.org for the latest
information and news.
Import an appliance into VirtualBox

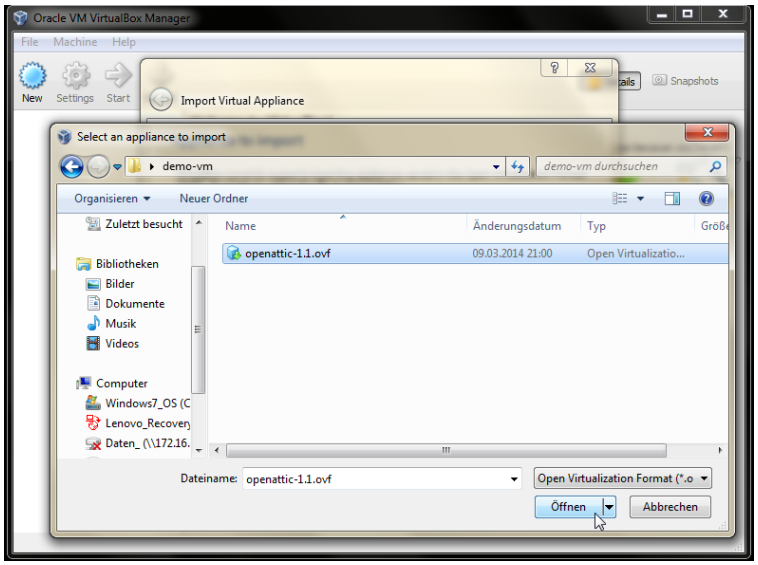
2. Import the appliance by clicking “Open appliance” and browsing to the directory in which you stored the downloaded openATTIC-demo VM. Select it and click “open”.

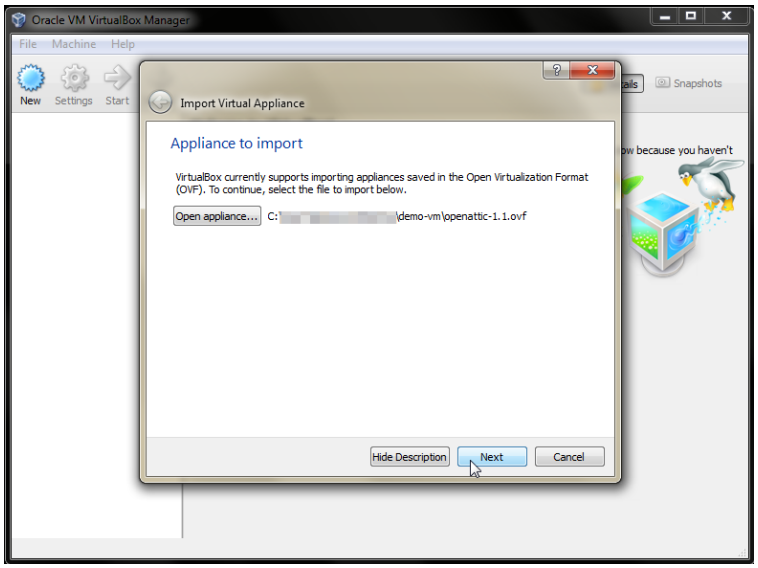


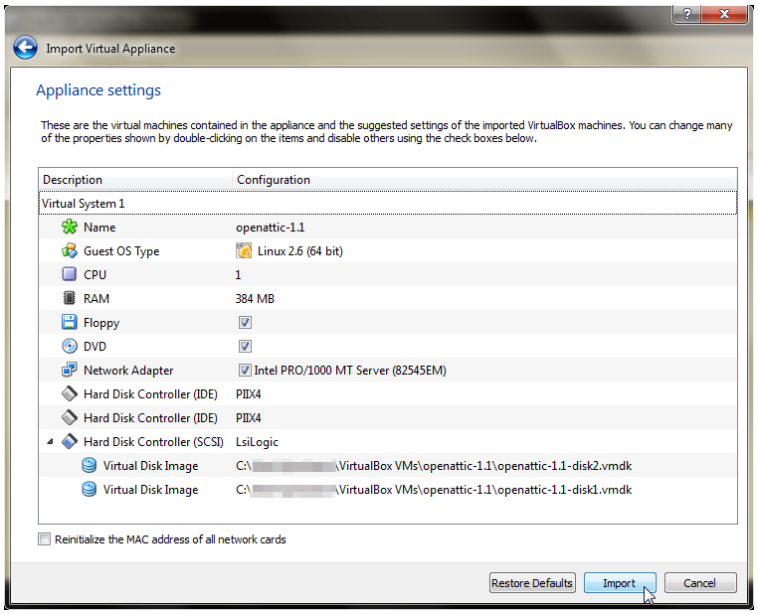
   Click “Next”

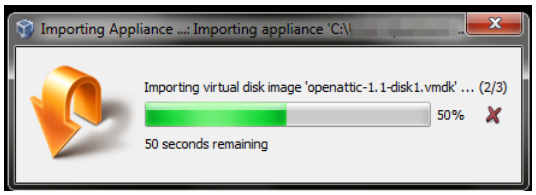
   Click “Import”

3. Importing the Appliance


Oracle VM VirtualBox Manager
File Machine Help
New Settings Start
Snapshots
Import Virtual Appliance
Select an appliance to import
demo-vm
demo-vm durchsuchen
Organisieren
Neuer Ordner
Zuletzt besucht
Bibliotheken
Bilder
Dokumente
Musik
Videos
Computer
Name
Änderungsdatum
Typ
openattic-1.1.ovf
09.03.2014 21:00
Open Virtualizatio...
Dateiname:
openattic-1.1.ovf
Open Virtualization Format (*.o
Öffnen
Abbrechen


Oracle VM VirtualBox Manager
File Machine Help
New Settings Start
Snapshots
Import Virtual Appliance
Appliance to import
VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below.
Open appliance...
C:\demo-vm\openattic-1.1.ovf
Hide Description
Next
Cancel




Importing Appliance ...: Importing appliance 'C:\
Importing virtual disk image 'openattic-1.1-disk1.vmdk' ... (2/3)
50%
50 seconds remaining

4. Starting the virtual machine

   Select the openattic-demo VM and click the "Start"-button above or right click the VM and select "Start"

5. Completion

   The system will boot automatically.

   openATTIC version 1.1: Here you can access the system with user `"root"` and password `"openattic"`.

   openATTIC version 1.2.: You can access the system with user `"root"` and password `"init"`.

   To access the openATTIC user interface in the browser you need either the ip-address or the hostname of the openATTIC host (just execute `ifconfig` or `hostname` in the commandline to get the required information)

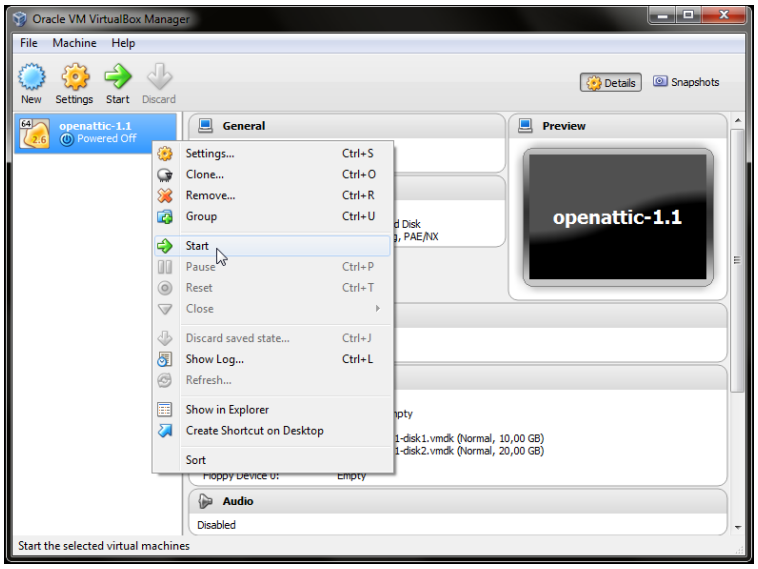
   Please login with user `"openattic"` and password `"openattic"`.

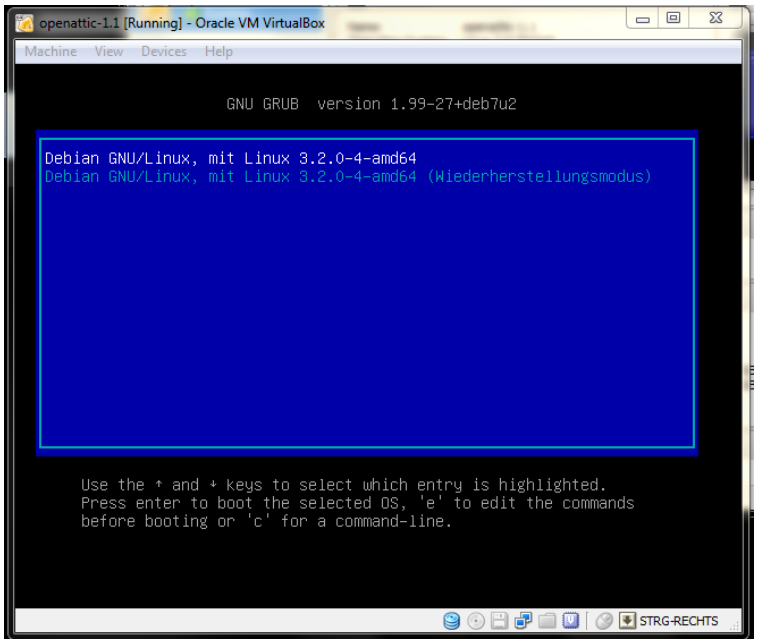
   openATTIC webinterface:

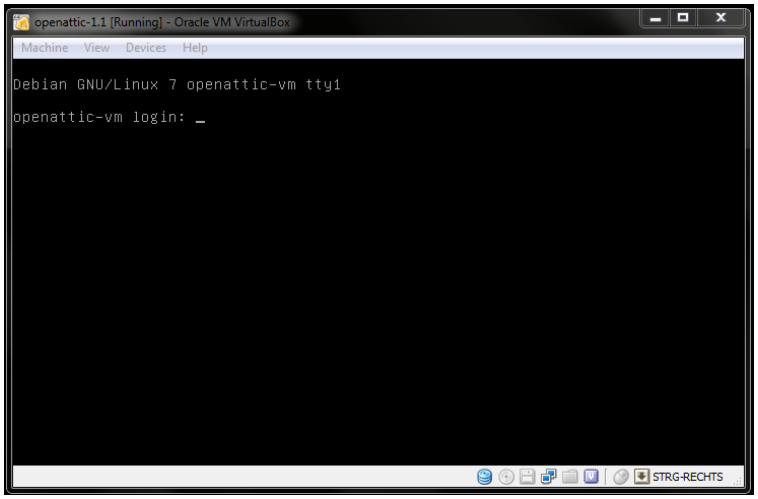
4.2.2 Demo VM for VMware

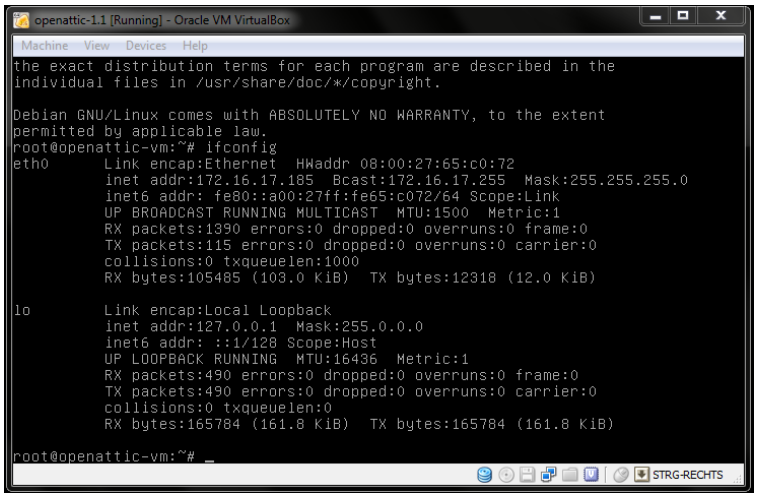
The openATTIC demo vm version 1.1 or 1.2 can be downloaded from http://apt.open-attic.org/vm/ and integrated into the VMware Workstation 9.0 as follows:

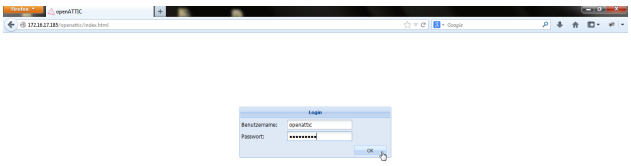
1. After opening the VMware Workstation 9.0 click on "Open a virtual machine"

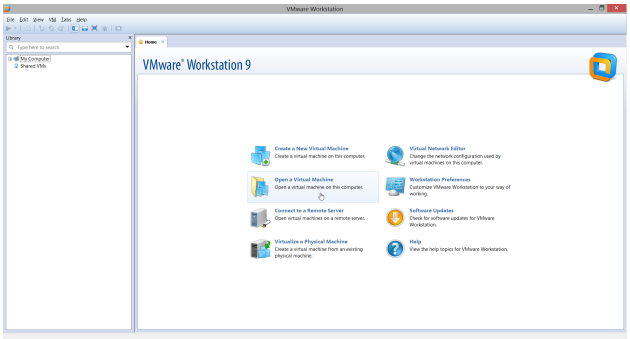

Oracle VM VirtualBox Manager
File Machine Help
New Settings Start Discard
Details Snapshots
openattic-1.1
Powered Off
General
Preview
Settings... Ctrl+S
Clone... Ctrl+O
Remove... Ctrl+R
Group Ctrl+U
Start
Pause Ctrl+P
Reset Ctrl+T
Close
Discard saved state... Ctrl+J
Show Log... Ctrl+L
Refresh...
Show in Explorer
Create Shortcut on Desktop
Sort
openattic-1.1
Audio
Disabled
Start the selected virtual machines


openattic-1.1 [Running] - Oracle VM VirtualBox
Machine View Devices Help
GNU GRUB version 1.99-27+deb7u2
Debian GNU/Linux, mit Linux 3.2.0-4-amd64
Debian GNU/Linux, mit Linux 3.2.0-4-amd64 (Wiederherstellungsmodus)
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the commands
before booting or 'c' for a command-line.
STRG-RECHTS

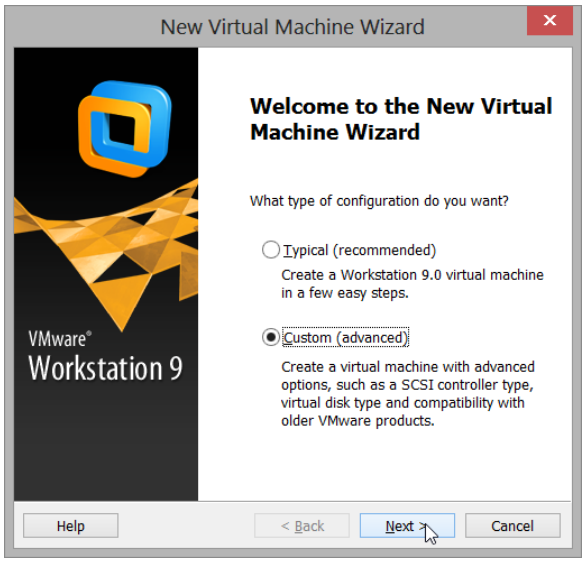

openattic-1.1 [Running] - Oracle VM VirtualBox
Machine View Devices Help
Debian GNU/Linux 7 openattic-vm tty1
openattic-vm login: _
STRG-RECHTS


openattic-1.1 [Running] - Oracle VM VirtualBox
Machine View Devices Help
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@openattic-vm:~# ifconfig
eth0 Link encap:Ethernet HWaddr 08:00:27:65:c0:72
inet addr:172.16.17.185 Bcast:172.16.17.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fe65:c072/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:1390 errors:0 dropped:0 overruns:0 frame:0
TX packets:115 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:105485 (103.0 KiB) TX bytes:12318 (12.0 KiB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:490 errors:0 dropped:0 overruns:0 frame:0
TX packets:490 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:165784 (161.8 KiB) TX bytes:165784 (161.8 KiB)
root@openattic-vm:~# _
STRG-RECHTS

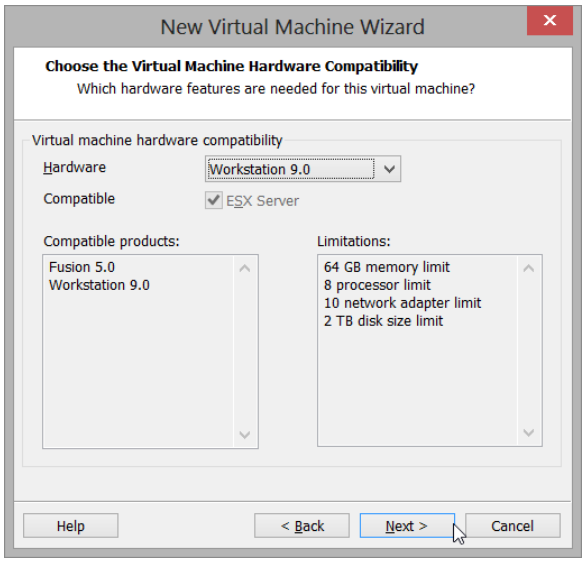



VMware Workstation
VMware® Workstation 9

2. Type of configuration - Choose the option “custom (advanced)”


New Virtual Machine Wizard
VMware
Workstation 9
Welcome to the New Virtual Machine Wizard
What type of configuration do you want?
Typical (recommended)
Create a Workstation 9.0 virtual machine in a few easy steps.
Custom (advanced)
Create a virtual machine with advanced options, such as a SCSI controller type, virtual disk type and compatibility with older VMware products.
Help
< Back
Next >
Cancel

3. Configuration of hardware compatibility

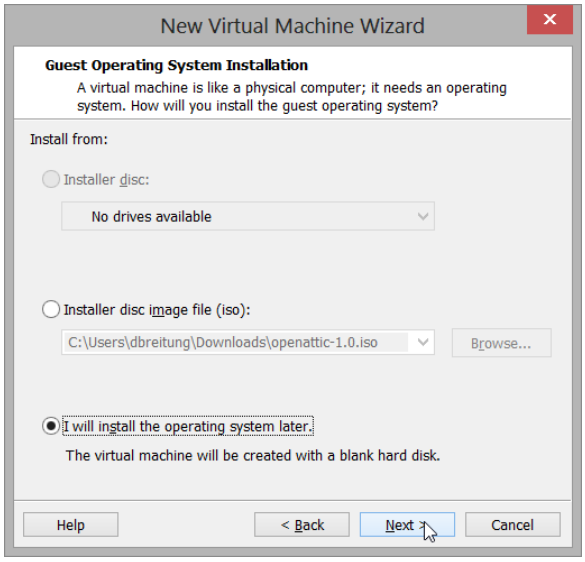


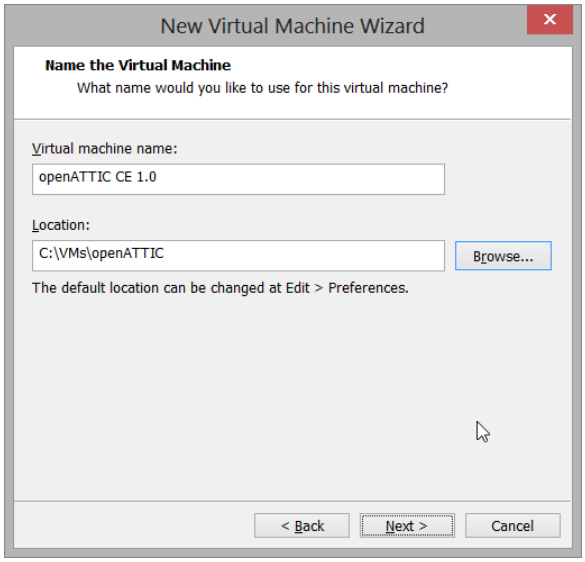
4. Guest operating system installation - Choose the option “I will install the operating system later”

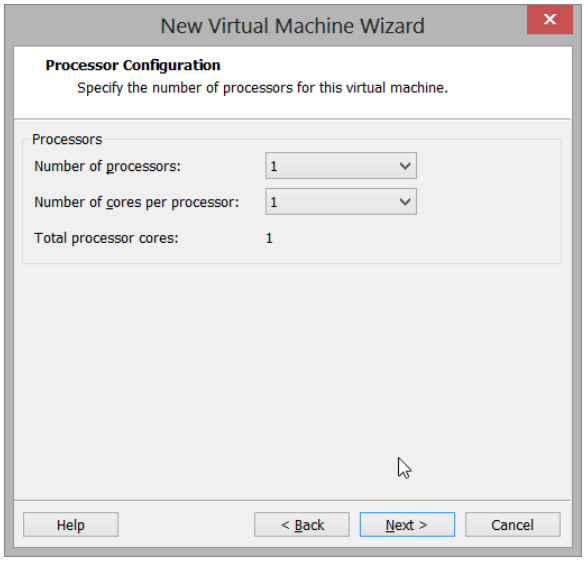
5. Choose “Linux” as operating system and version “Debian 6 64-bit”

6. If necessary the virtual machine name and the location can be adjusted here

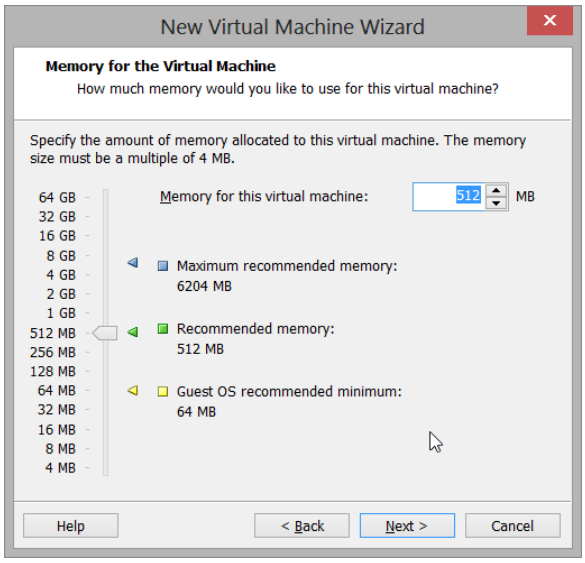
7. VM processor configuration


New Virtual Machine Wizard
Guest Operating System Installation
A virtual machine is like a physical computer; it needs an operating system. How will you install the guest operating system?
Install from:
Installer disc:
No drives available
Installer disc image file (iso):
C:\Users\dbreitung\Downloads\openattic-1.0.iso
Browse...
I will install the operating system later.
The virtual machine will be created with a blank hard disk.
Help
< Back
Next >
Cancel

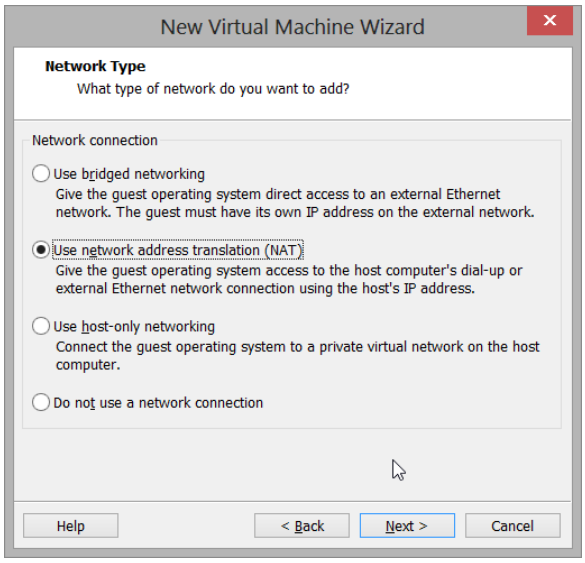

New Virtual Machine Wizard
Name the Virtual Machine
What name would you like to use for this virtual machine?
Virtual machine name:
openATTIC CE 1.0
Location:
C:\VMs\openATTIC
Browse...
The default location can be changed at Edit > Preferences.
< Back
Next >
Cancel


New Virtual Machine Wizard
Processor Configuration
Specify the number of processors for this virtual machine.
Processors
Number of processors:
1
Number of cores per processor:
1
Total processor cores:
1
Help
< Back
Next >
Cancel

8. VM RAM configuration



9. Network type - Choose “Use network address translation (NAT)”

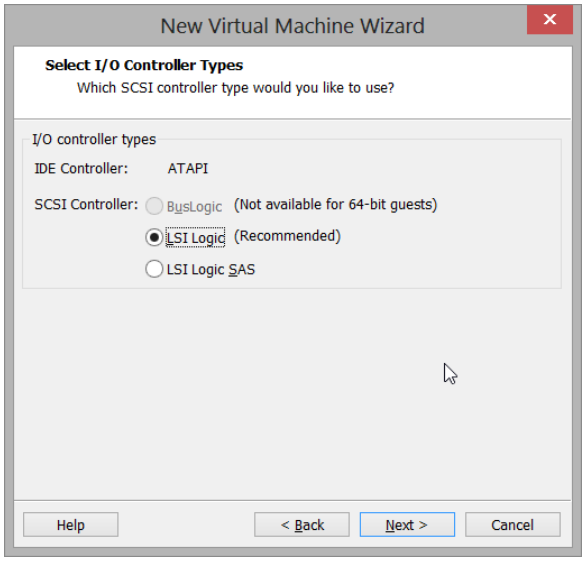

New Virtual Machine Wizard
Network Type
What type of network do you want to add?
Network connection
Use bridged networking
Give the guest operating system direct access to an external Ethernet network. The guest must have its own IP address on the external network.
Use network address translation (NAT)
Give the guest operating system access to the host computer's dial-up or external Ethernet network connection using the host's IP address.
Use host-only networking
Connect the guest operating system to a private virtual network on the host computer.
Do not use a network connection
Help
< Back
Next >
Cancel

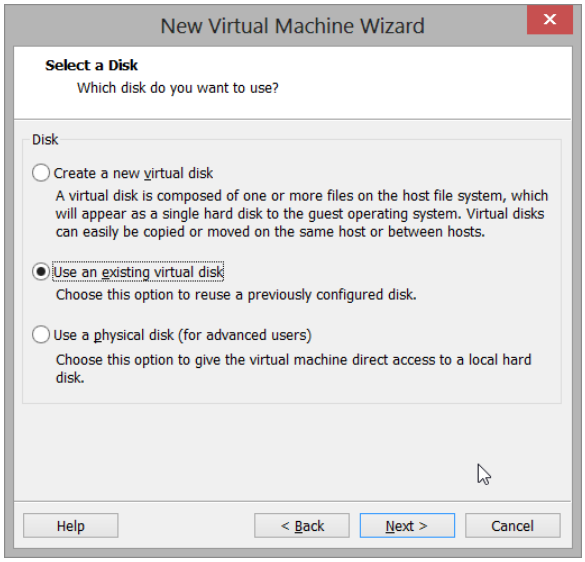
10. I/O controller types - Choose the recommended option “LSI Logic”

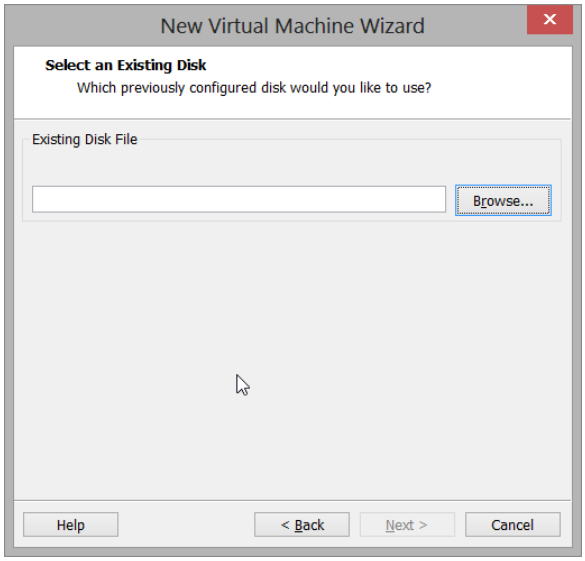
11. Select a disk - Choose “Use an existing virtual disk”

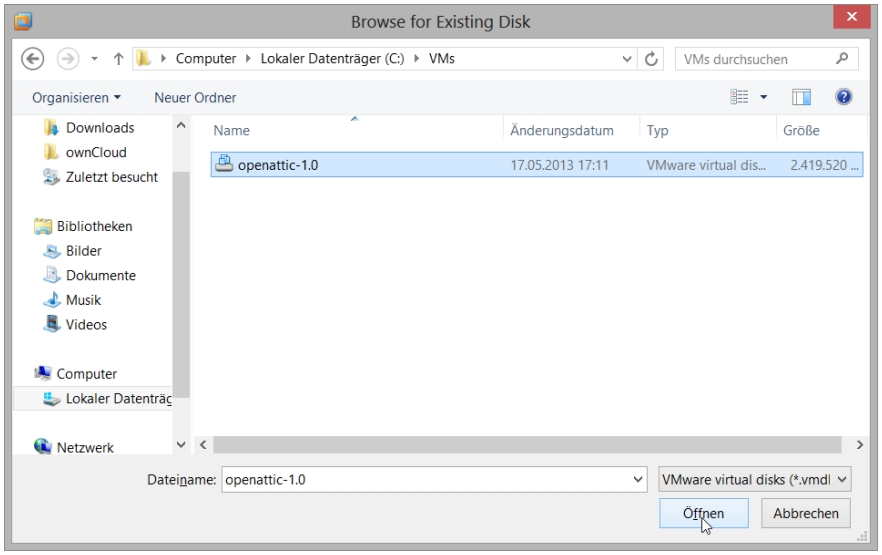
12.

13. Choose the .vmdk file

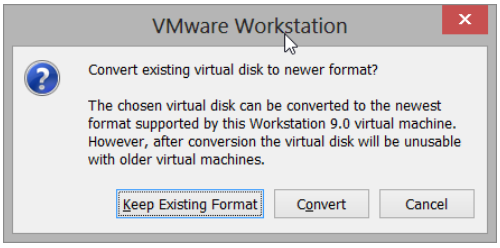

New Virtual Machine Wizard
Select I/O Controller Types
Which SCSI controller type would you like to use?
I/O controller types
IDE Controller: ATAPI
SCSI Controller: BusLogic (Not available for 64-bit guests)
LSI Logic (Recommended)
LSI Logic SAS
Help
< Back
Next >
Cancel


New Virtual Machine Wizard
Select a Disk
Which disk do you want to use?
Disk
Create a new virtual disk
A virtual disk is composed of one or more files on the host file system, which will appear as a single hard disk to the guest operating system. Virtual disks can easily be copied or moved on the same host or between hosts.
Use an existing virtual disk
Choose this option to reuse a previously configured disk.
Use a physical disk (for advanced users)
Choose this option to give the virtual machine direct access to a local hard disk.
Help
< Back
Next >
Cancel


New Virtual Machine Wizard
Select an Existing Disk
Which previously configured disk would you like to use?
Existing Disk File
Browse...
Help
< Back
Next >
Cancel



14. Here you can keep the existing format


VMware Workstation
Convert existing virtual disk to newer format?
The chosen virtual disk can be converted to the newest format supported by this Workstation 9.0 virtual machine. However, after conversion the virtual disk will be unusable with older virtual machines.
Keep Existing Format
Convert
Cancel

15. Click the finish button

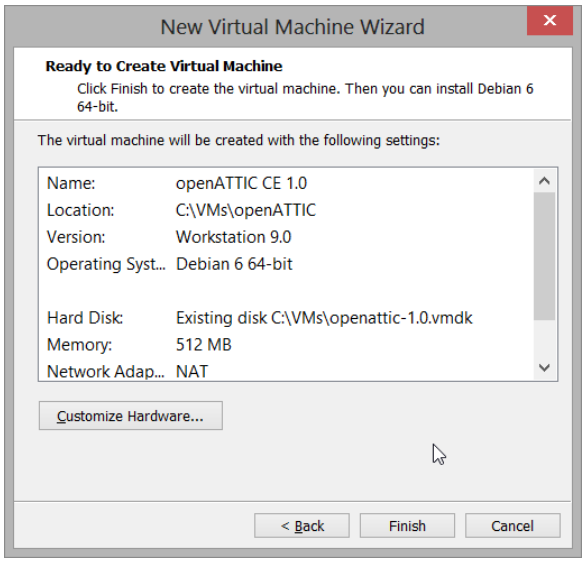
16. Then you can start the virtual machine

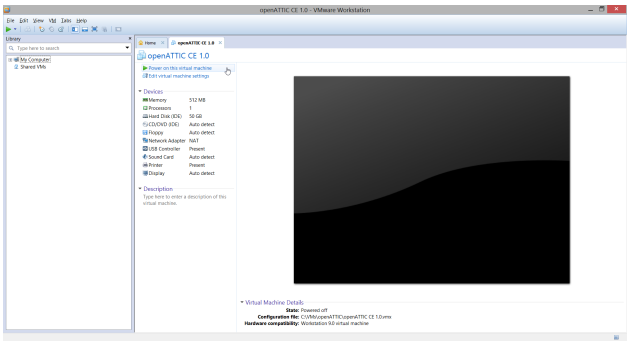
17. **openATTIC version 1.1**: Here you can access the openATTIC system via console by entering `"root"` as user and `"openattic"` as password.

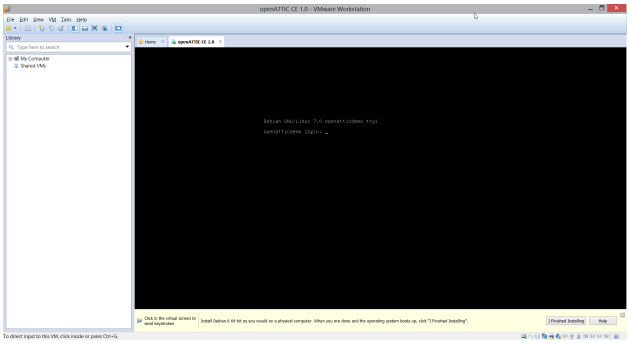
    openATTIC version 1.2: Login with user `"root"` and password `"init"`.

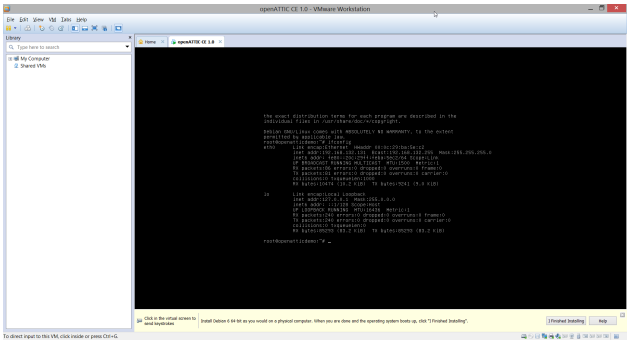
18. By typing the “ifconfig” command you will get the ip address of the demo-vm.

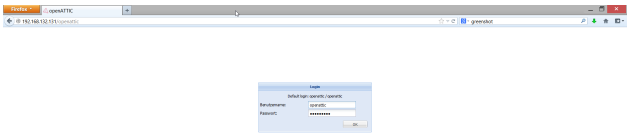
19. To access the openATTIC user interface you can login with user `"openattic"` and password `"openattic"`


New Virtual Machine Wizard
Ready to Create Virtual Machine
Click Finish to create the virtual machine. Then you can install Debian 6 64-bit.
The virtual machine will be created with the following settings:
Name: openATTIC CE 1.0
Location: C:\VMs\openATTIC
Version: Workstation 9.0
Operating Syst... Debian 6 64-bit
Hard Disk: Existing disk C:\VMs\openattic-1.0.vmdk
Memory: 512 MB
Network Adap... NAT
Customize Hardware...
< Back
Finish
Cancel









4.3 Step by step installation guides

This section will guide you through the different installation types step by step. If you already have a running ubuntu or debian system you might want to take a look at the :ref: *quick start <quickstart_guides_install>* section.

4.3.1 ISO installation in graphical mode

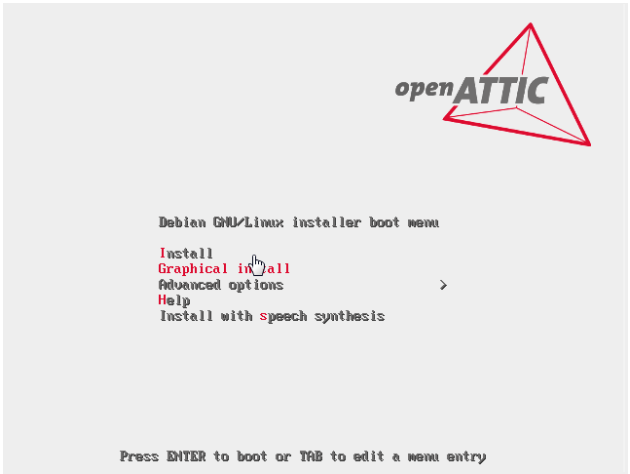
The openATTIC iso file for virtualbox/VMware Workstation can be downloaded from www.openattic.org/downloads.html

Available versions:

```
openattic-1.0.iso               openATTIC Stable v1.0
openattic-1.1.iso               openATTIC Stable v1.1
openattic-weekly.iso    openATTIC Weekly Build
```

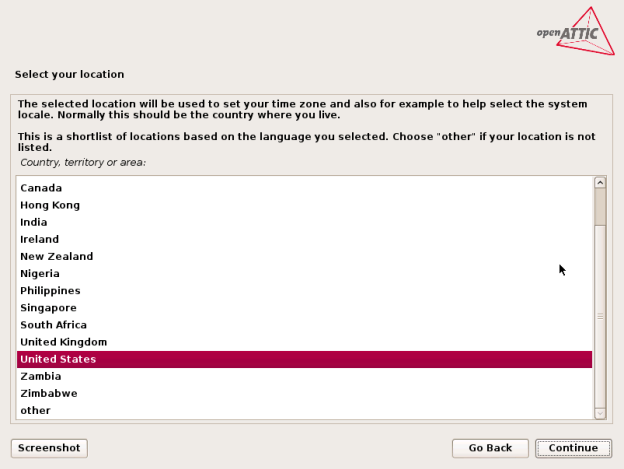
This guide is for the installation using the openATTIC ISO image in graphical mode.

The following screenshots will guide you step by step through the installation. After booting from the CD, you will see the following screen:



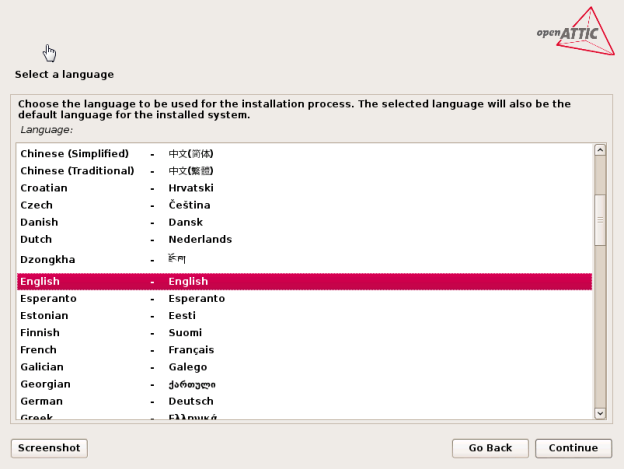
You can choose between the different installation options. To run the installation in graphical mode choose `Graphical Install`.

1. Language configuration



System language configuration

Set the time zone:



Choose keyboard layout:

2. Network configuration

   Choose between configuring the network manually or using auto-configure

   Waiting time (in seconds) for link detection:

3. System configuration

   Create the hostname

   Enter the Domain name if available - else leave it empty

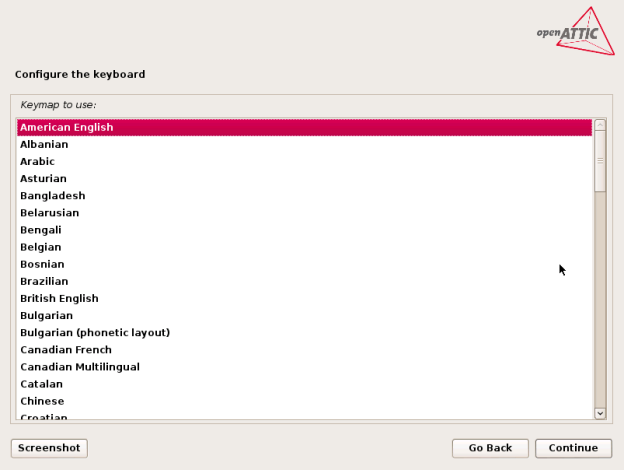
   Insert a root password - re-enter password to verify

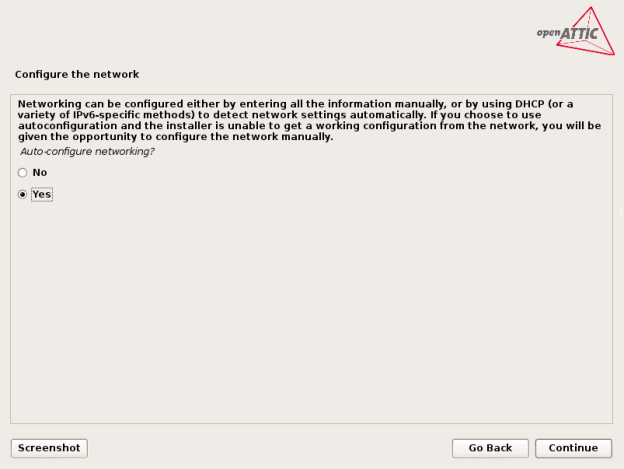
4. Partitioning

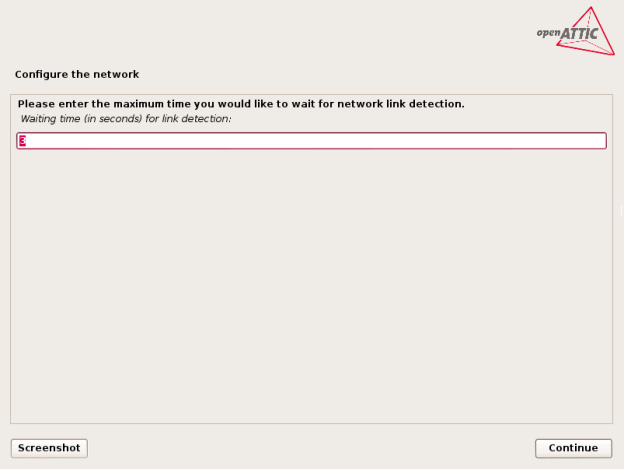
   Here we choose the the first option - “use entire disk”

   Choose the disk where the system should be installed

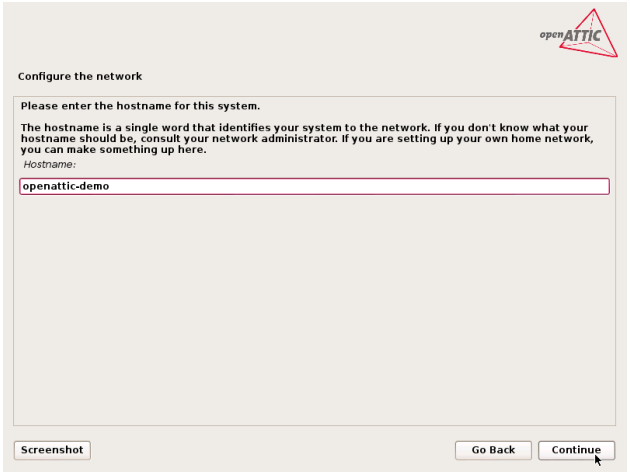
   Partition scheme - “All files in one partition” is the standard and adequate


Configure the keyboard
Keymap to use:
American English
Albanian
Arabic
Asturian
Bangladesh
Belarusian
Bengali
Belgian
Bosnian
Brazilian
British English
Bulgarian
Bulgarian (phonetic layout)
Canadian French
Canadian Multilingual
Catalan
Chinese
Screenshot
Go Back
Continue

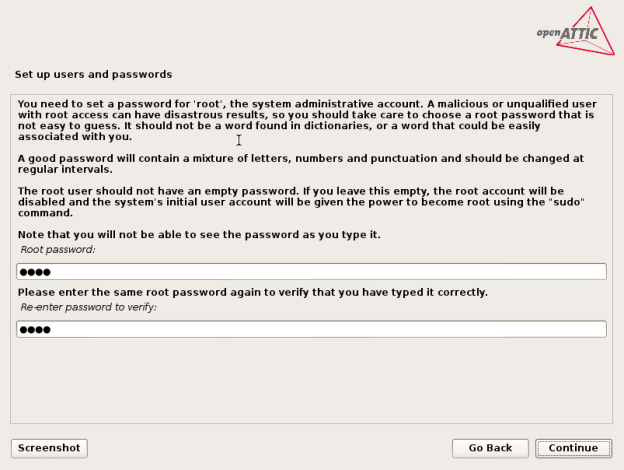

Configure the network
Networking can be configured either by entering all the information manually, or by using DHCP (or a variety of IPv6-specific methods) to detect network settings automatically. If you choose to use autoconfiguration and the installer is unable to get a working configuration from the network, you will be given the opportunity to configure the network manually.
Auto-configure networking?
No
Yes
Screenshot
Go Back
Continue

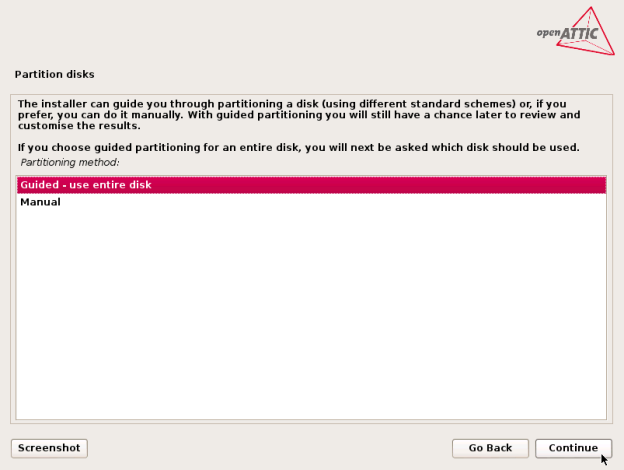

Configure the network
Please enter the maximum time you would like to wait for network link detection.
Waiting time (in seconds) for link detection:
Screenshot
Continue

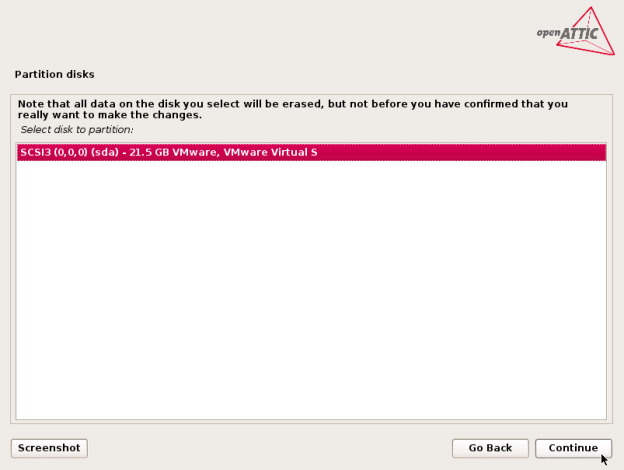
Configure the network
Attempting IPv6 autoconfiguration...
Cancel

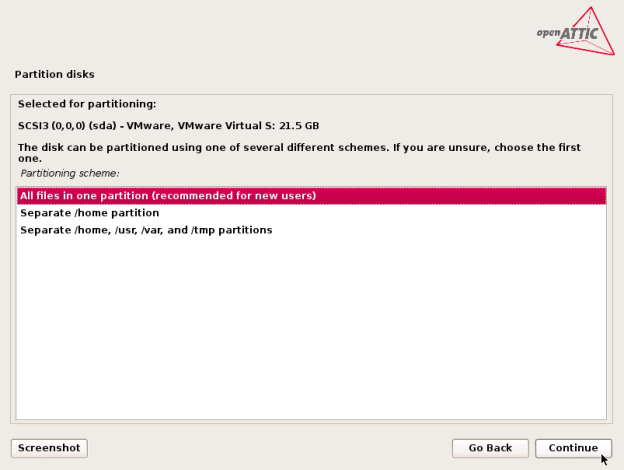

Configure the network
Please enter the hostname for this system.
The hostname is a single word that identifies your system to the network. If you don't know what your hostname should be, consult your network administrator. If you are setting up your own home network, you can make something up here.
Hostname:
openattic-demo
Screenshot
Go Back
Continue

Configure the network
The domain name is the part of your Internet address to the right of your host name. It is often something that ends in .com, .net, .edu, or .org. If you are setting up a home network, you can make something up, but make sure you use the same domain name on all your computers.
Domain name:
example.local
Screenshot
Go Back
Continue

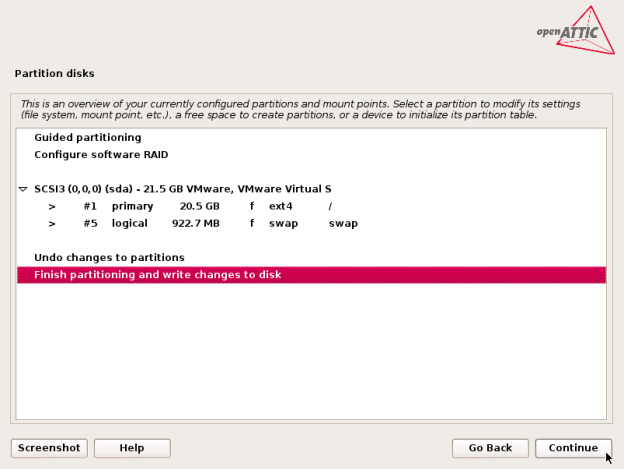

Set up users and passwords
You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.
A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.
Note that you will not be able to see the password as you type it.
Root password:
Please enter the same root password again to verify that you have typed it correctly.
Re-enter password to verify:
Screenshot
Go Back
Continue


Partition disks
The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.
If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.
Partitioning method:
Guided - use entire disk
Manual
Screenshot
Go Back
Continue


Partition disks
Note that all data on the disk you select will be erased, but not before you have confirmed that you really want to make the changes.
Select disk to partition:
SCSI3 (0,0,0) (sda) - 21.5 GB VMware, VMware Virtual S
Screenshot
Go Back
Continue


Partition disks
Selected for partitioning:
SCSI3 (0,0,0) (sda) - VMware, VMware Virtual S: 21.5 GB
The disk can be partitioned using one of several different schemes. If you are unsure, choose the first one.
Partitioning scheme:
All files in one partition (recommended for new users)
Separate /home partition
Separate /home, /usr, /var, and /tmp partitions
Screenshot
Go Back
Continue

Apply the partitioning


Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
Configure software RAID
SCSI3 (0,0,0) (sda) - 21.5 GB VMware, VMware Virtual S
> #1 primary 20.5 GB f ext4 /
> #5 logical 922.7 MB f swap swap
Undo changes to partitions
Finish partitioning and write changes to disk
Screenshot
Help
Go Back
Continue

Write the changes to disk

Installing the base system

5. Kerberos

   Here you can enter the Kerberos Realm. If you don't want to configure Kerberos you can skip this part by leaving the field empty and clicking "Continue". This is an example configuration for Kerberos.

   Enter the Kerberos server (general case: domaincontroller) and the administrative server for your Kerberos realm

6. Package manager

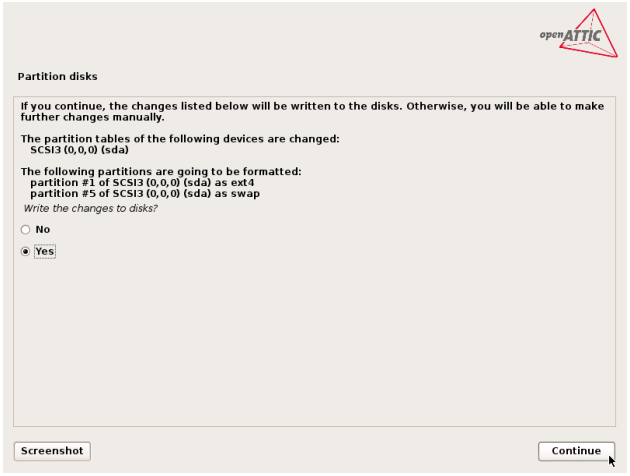
   Choose the protocol for file downloads

   Choose the country for the package manager

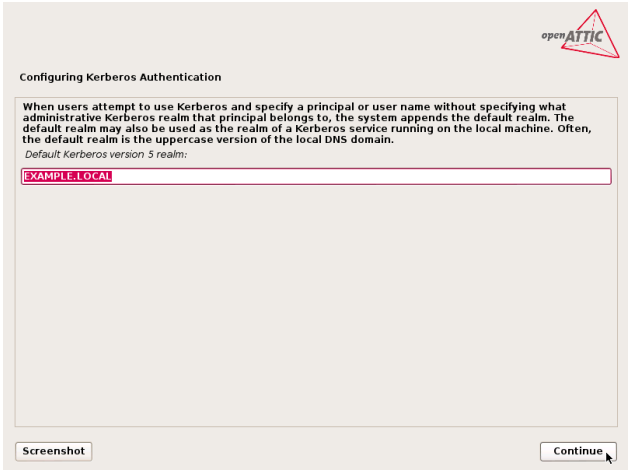
   Select the mirror server

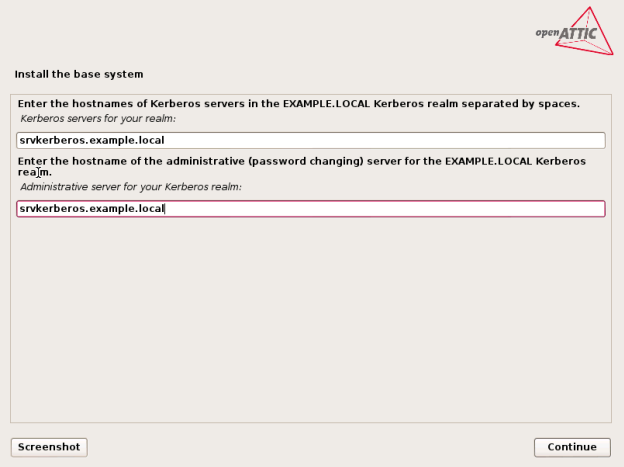
   Enter HTTP proxy information (leave blank for none)

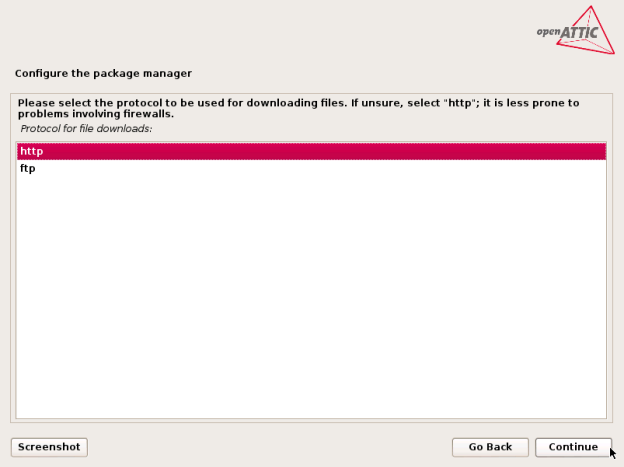
   Configuring apt

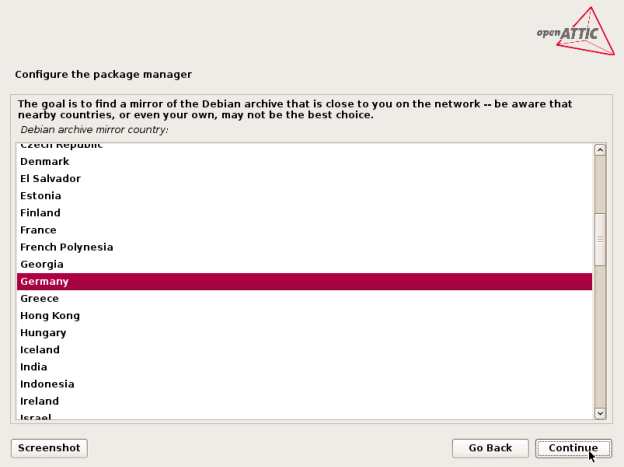

Partition disks
If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.
The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)
The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as ext4
partition #5 of SCSI3 (0,0,0) (sda) as swap
Write the changes to disks?
No
Yes
Screenshot
Continue

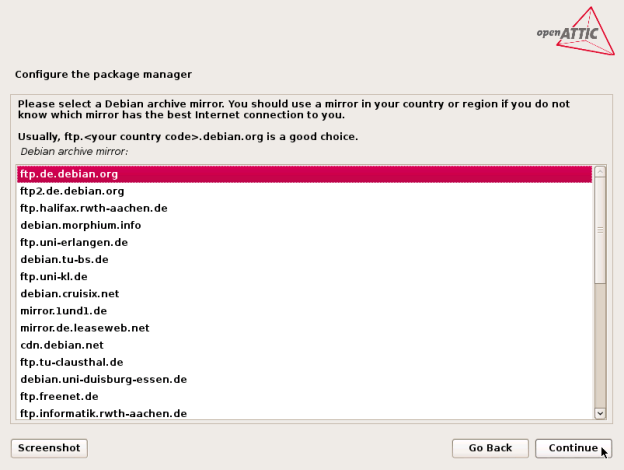
Install the base system
Installing the base system
Retrieving mount...

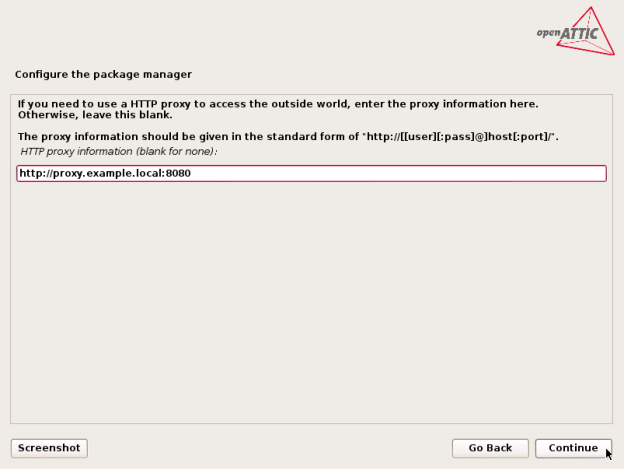

Configuring Kerberos Authentication
When users attempt to use Kerberos and specify a principal or user name without specifying what administrative Kerberos realm that principal belongs to, the system appends the default realm. The default realm may also be used as the realm of a Kerberos service running on the local machine. Often, the default realm is the uppercase version of the local DNS domain.
Default Kerberos version 5 realm:
EXAMPLE.LOCAL
Screenshot
Continue


Install the base system
Enter the hostnames of Kerberos servers in the EXAMPLE.LOCAL Kerberos realm separated by spaces.
Kerberos servers for your realm:
srvkerberos.example.local
Enter the hostname of the administrative (password changing) server for the EXAMPLE.LOCAL Kerberos realm.
Administrative server for your Kerberos realm:
srvkerberos.example.local
Screenshot
Continue

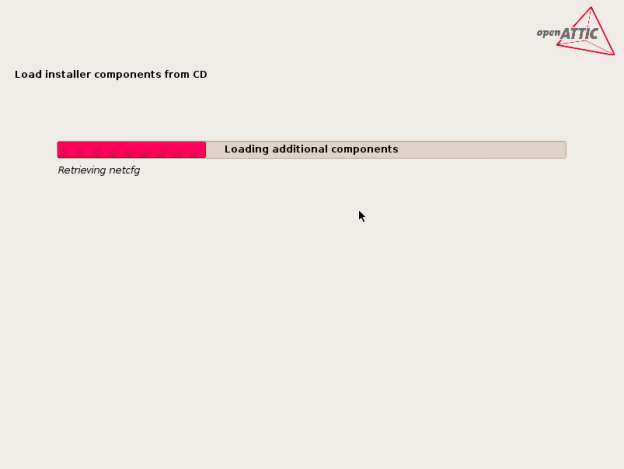

Configure the package manager
Please select the protocol to be used for downloading files. If unsure, select "http"; it is less prone to problems involving firewalls.
Protocol for file downloads:
http
ftp
Screenshot
Go Back
Continue


Configure the package manager
The goal is to find a mirror of the Debian archive that is close to you on the network -- be aware that nearby countries, or even your own, may not be the best choice.
Debian archive mirror country:
Denmark
El Salvador
Estonia
Finland
France
French Polynesia
Georgia
Germany
Greece
Hong Kong
Hungary
Iceland
India
Indonesia
Ireland
Screenshot
Go Back
Continue


Configure the package manager
Please select a Debian archive mirror. You should use a mirror in your country or region if you do not know which mirror has the best Internet connection to you.
Usually, ftp.<your country code>.debian.org is a good choice.
Debian archive mirror:
ftp.de.debian.org
ftp2.de.debian.org
ftp.halifax.rwth-aachen.de
debian.morphium.info
ftp.uni-erlangen.de
debian.tu-bs.de
ftp.uni-kl.de
debian.cruisix.net
mirror.1und1.de
mirror.de.leaseweb.net
cdn.debian.net
ftp.tu-clausthal.de
debian.uni-duisburg-essen.de
ftp.freenet.de
ftp.informatik.rwth-aachen.de
Screenshot
Go Back
Continue

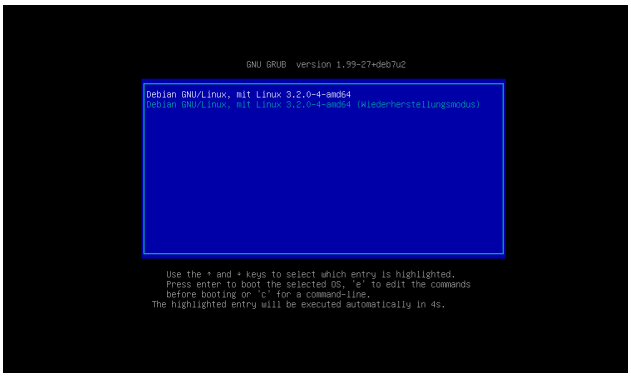

Configure the package manager
If you need to use a HTTP proxy to access the outside world, enter the proxy information here. Otherwise, leave this blank.
The proxy information should be given in the standard form of "http://[[user][:pass]@]host[:port]/".
HTTP proxy information (blank for none):
http://proxy.example.local:8080
Screenshot
Go Back
Continue

Configure the package manager
Configuring apt
Scanning the mirror...
Cancel



7. Completion

   After completing the installation the system will boot automatically. The system is based on Debian Wheezy kernel 3.2


GNU GRUB version 1.99-27+deb7u2
Debian GNU/Linux, mit Linux 3.2.0-4-amd64
Debian GNU/Linux, mit Linux 3.2.0-4-amd64 (Wiederherstellungsmodus)
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line.
The highlighted entry will be executed automatically in 4s.

   Now you can see the login-prompt with a link where you can access openATTIC via the browser. Please login as user "`openattic`" and password "`openattic`".

4.3.2 ISO installation in text mode

The openATTIC iso file for virtualbox/VMware Workstation can be downloaded from http://www.openattic.org/downloads.html or http://apt.openattic.org/iso

Available versions:

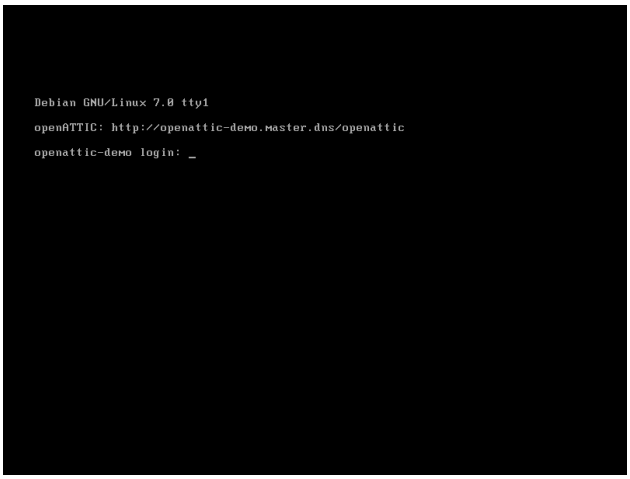
```
openattic-1.0.iso              openATTIC Stable v1.0
openattic-1.1.iso              openATTIC Stable v1.1
openattic-weekly.iso           openATTIC Weekly Build
```

This guide is for the installation using the openATTIC ISO image in text mode.

The following screenshots will guide you step-by-step through the installation. After booting from the CD, you will see the following screen:

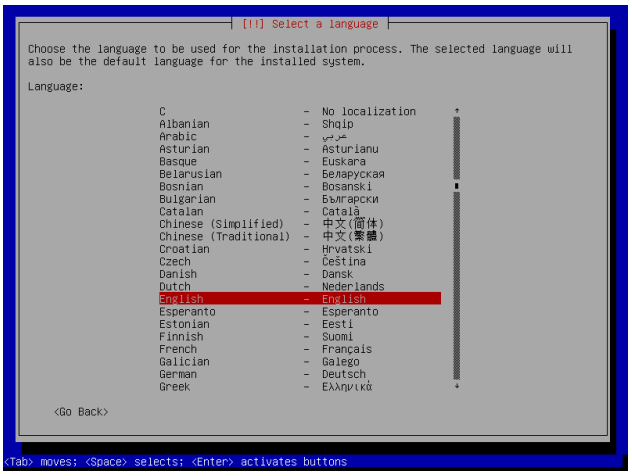
You can choose between the different installation options. To run the installation in text mode choose `Install`.

1. Language configuration

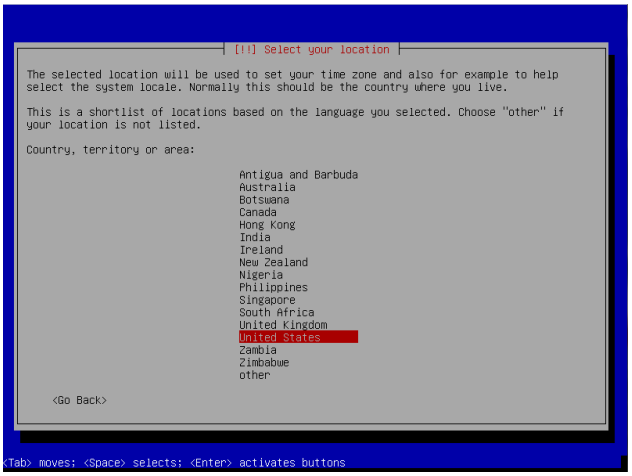

Debian GNU/Linux 7.0 tty1
openATTIC: http://openattic-demo.master.dns/openattic
openattic-demo login: _

openATTIC
Debian GNU/Linux installer boot menu
Install
Graphical install
Advanced options
Help
Install with speech synthesis
Press ENTER to boot or TAB to edit a menu entry

System language configuration



Set the time zone:



Choose keyboard layout:

Required components will be loaded automatically.

2. Network configuration

   Choose between configuring the network manually or using auto-configure.

   Waiting time (in seconds) for link detection:

3. System configuration

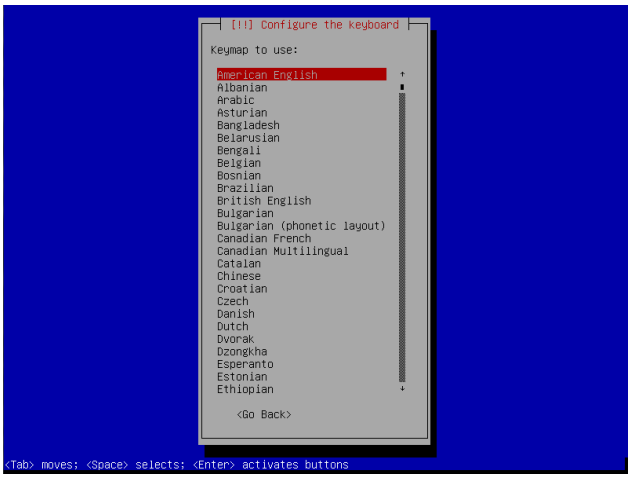
   Create the hostname

   Enter the domain name if available - otherwise leave it empty

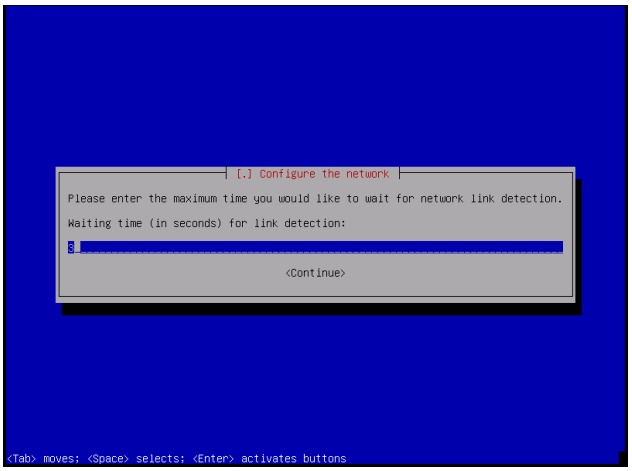
   Insert a root password

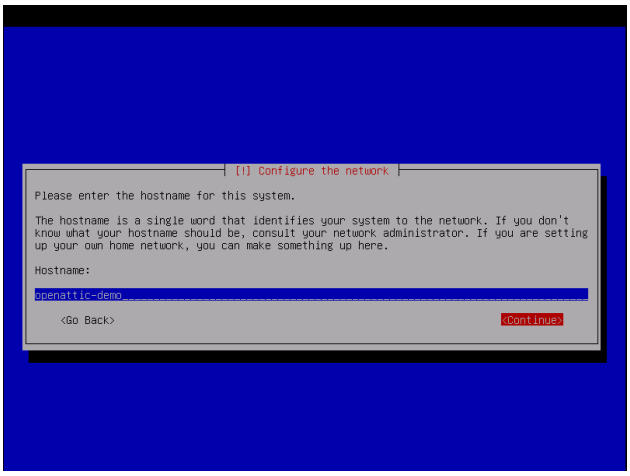
   Re-enter password to verify

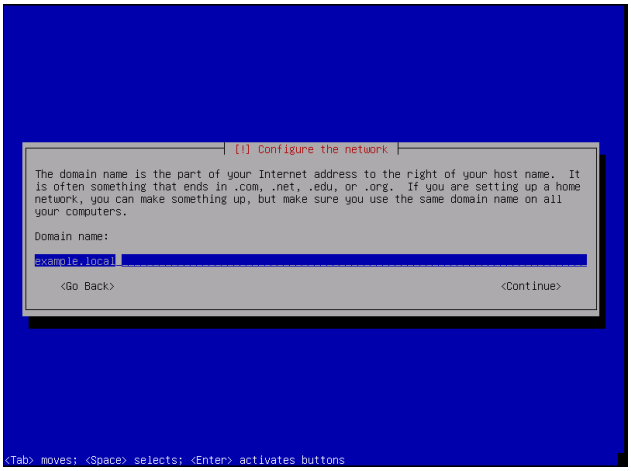
4. Partitioning

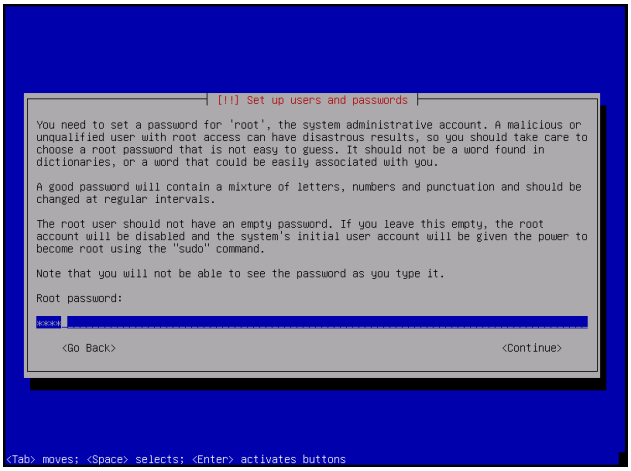

[!!] Configure the keyboard
Keymap to use:
American English
Albanian
Arabic
Asturian
Bangladesh
Belarusian
Bengali
Belgian
Bosnian
Brazilian
British English
Bulgarian
Bulgarian (phonetic layout)
Canadian French
Canadian Multilingual
Catalan
Chinese
Croatian
Czech
Danish
Dutch
Dvorak
Dzongkha
Esperanto
Estonian
Ethiopian
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons

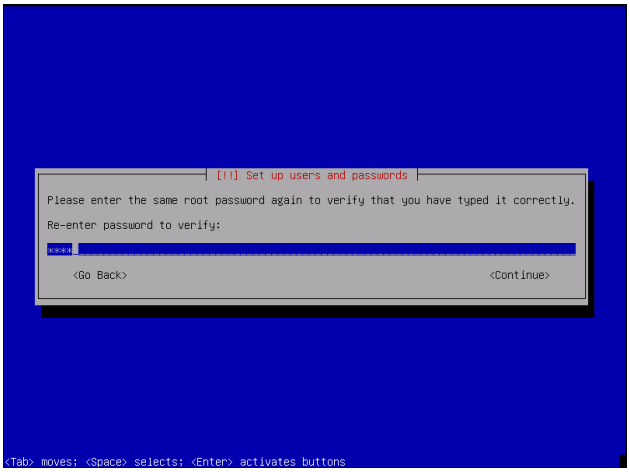
[?] Configure the network
Networking can be configured either by entering all the information manually, or by using DHCP (or a variety of IPv6-specific methods) to detect network settings automatically. If you choose to use autoconfiguration and the installer is unable to get a working configuration from the network, you will be given the opportunity to configure the network manually.
Auto-configure networking?
<Go Back>
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons

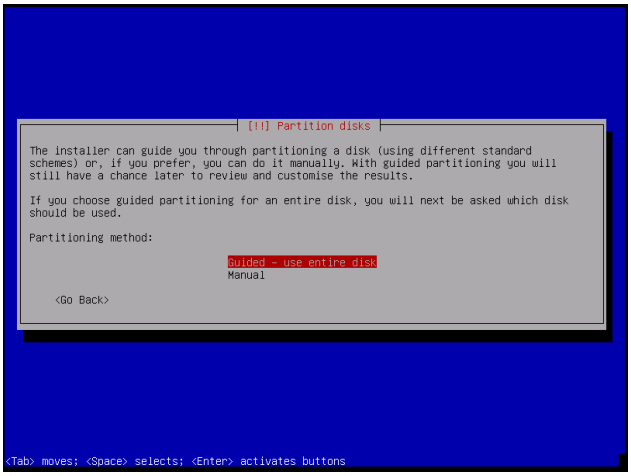

[.] Configure the network
Please enter the maximum time you would like to wait for network link detection.
Waiting time (in seconds) for link detection:
0
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons


[!] Configure the network
Please enter the hostname for this system.
The hostname is a single word that identifies your system to the network. If you don't know what your hostname should be, consult your network administrator. If you are setting up your own home network, you can make something up here.
Hostname:
openattic-demo
<Go Back>
<Continue>


[!] Configure the network
The domain name is the part of your Internet address to the right of your host name. It is often something that ends in .com, .net, .edu, or .org. If you are setting up a home network, you can make something up, but make sure you use the same domain name on all your computers.
Domain name:
example.local
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

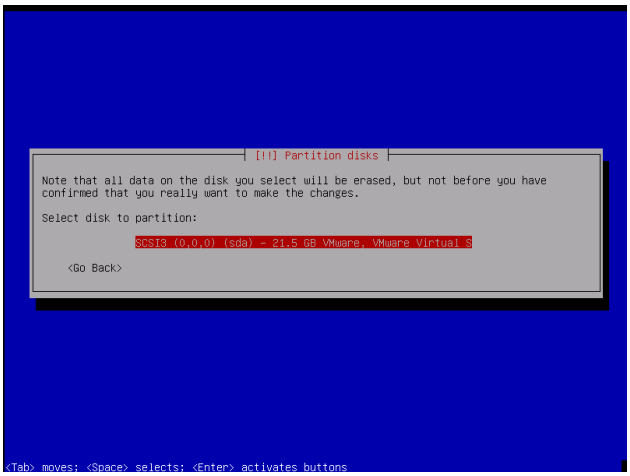

[!!] Set up users and passwords
You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.
A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.
Note that you will not be able to see the password as you type it.
Root password:
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons


[!!] Set up users and passwords
Please enter the same root password again to verify that you have typed it correctly.
Re-enter password to verify:
****
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

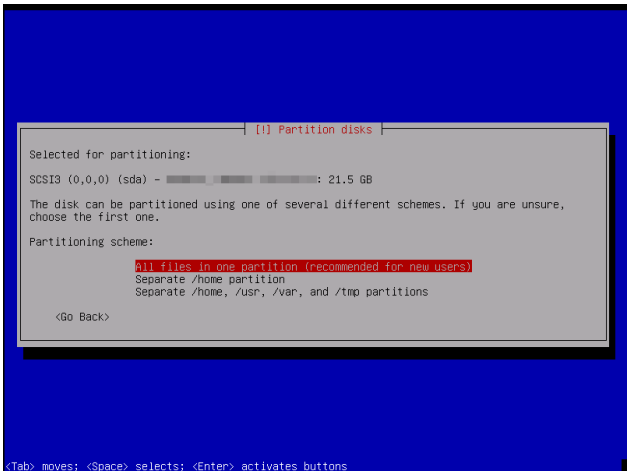

[!!] Partition disks
The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.
If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.
Partitioning method:
Guided - use entire disk
Manual
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons

Here we choose the first option - “use entire disk”

Choose the disk where the system should be installed


[!!] Partition disks
Note that all data on the disk you select will be erased, but not before you have confirmed that you really want to make the changes.
Select disk to partition:
SCSI3 (0,0,0) (sda) - 21.5 GB VMware, VMware Virtual S
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons

Partition scheme - “All files in one partition” is standard and adequate for most cases


[!] Partition disks
Selected for partitioning:
SCSI3 (0,0,0) (sda) - : 21.5 GB
The disk can be partitioned using one of several different schemes. If you are unsure, choose the first one.
Partitioning scheme:
All files in one partition (recommended for new users)
Separate /home partition
Separate /home, /usr, /var, and /tmp partitions
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons

Apply the partitioning

Write the changes to disk

Install the base system

5. Kerberos

   Here you can enter the Kerberos Realm. If you don’t want to configure Kerberos you can skip this part by leaving the field empty and clicking “Continue”. This is an example configuration for Kerberos.

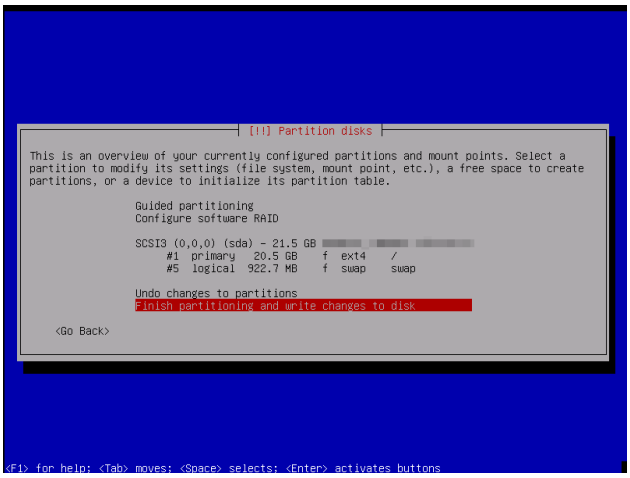
   Enter the Kerberos server - general case is a domaincontroller

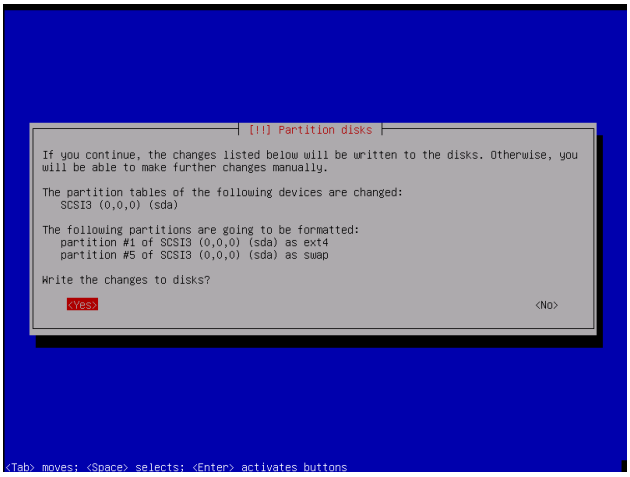
   Enter the administrative server for your Kerberos realm

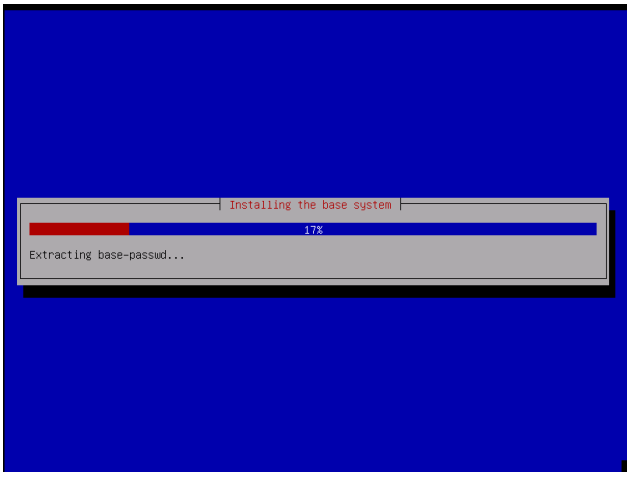
6. Package manager

   Choose the protocol for file downloads

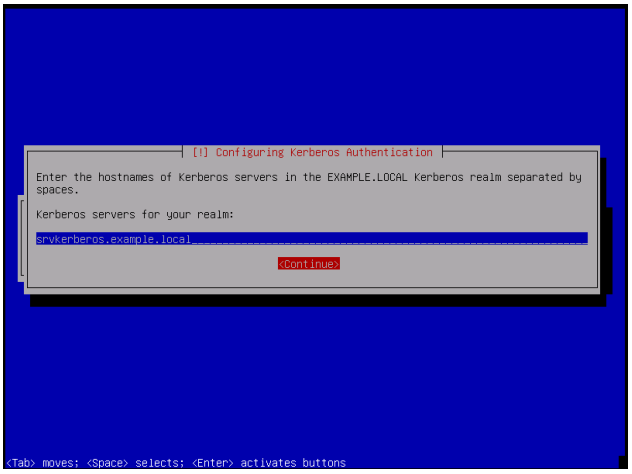
   Choose the country for the package manager

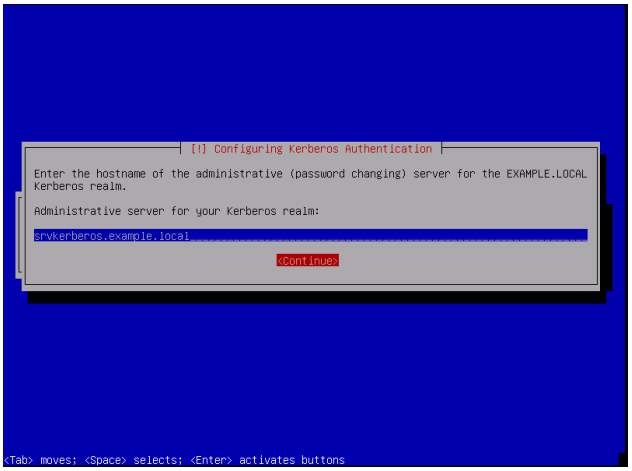

[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
Configure software RAID
SCSI3 (0,0,0) (sda) - 21.5 GB
#1 primary 20.5 GB f ext4 /
#5 logical 922.7 MB f swap swap
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons

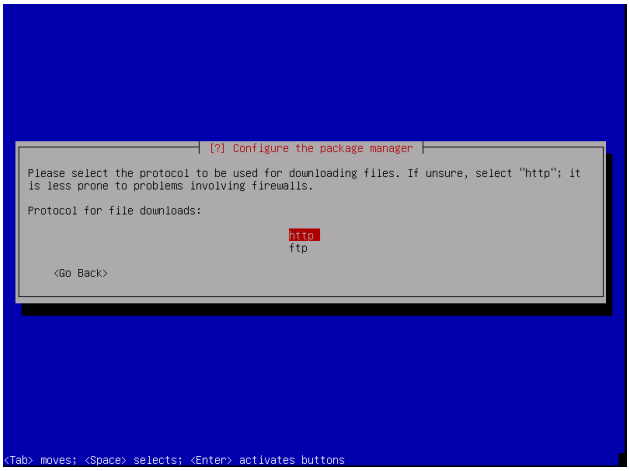

[!!] Partition disks
If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.
The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)
The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as ext4
partition #5 of SCSI3 (0,0,0) (sda) as swap
Write the changes to disks?
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons

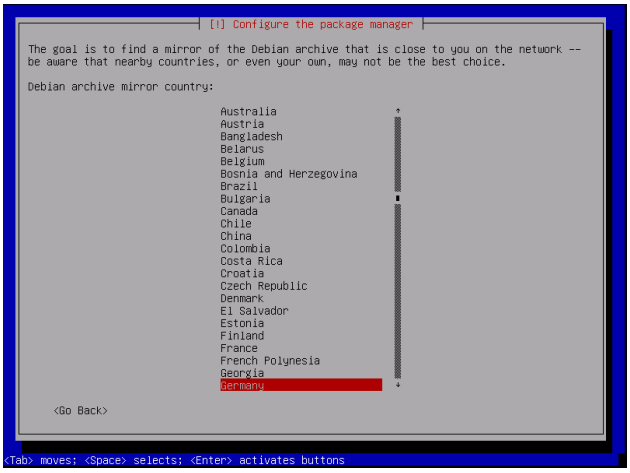

Installing the base system
17%
Extracting base-passwd...

[!!] Configuring Kerberos Authentication
When users attempt to use Kerberos and specify a principal or user name without specifying what administrative Kerberos realm that principal belongs to, the system appends the default realm. The default realm may also be used as the realm of a Kerberos service running on the local machine. Often, the default realm is the uppercase version of the local DNS domain.
Default Kerberos version 5 realm:
EXAMPLE.LOCAL
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

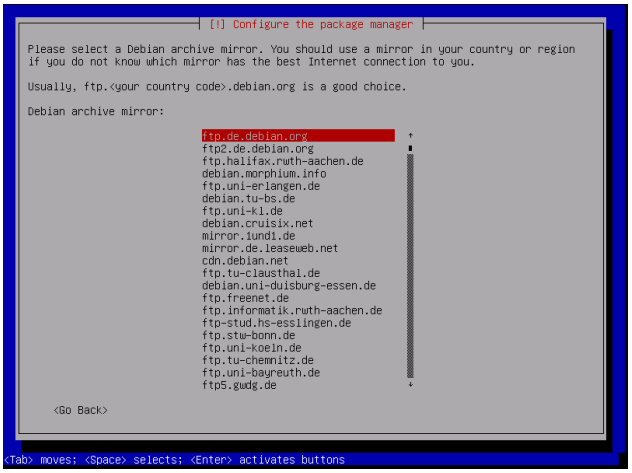

[!] Configuring Kerberos Authentication
Enter the hostnames of Kerberos servers in the EXAMPLE.LOCAL Kerberos realm separated by spaces.
Kerberos servers for your realm:
srvkerberos.example.local
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons


[!] Configuring Kerberos Authentication
Enter the hostname of the administrative (password changing) server for the EXAMPLE.LOCAL Kerberos realm.
Administrative server for your Kerberos realm:
srvkerberos.example.local
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

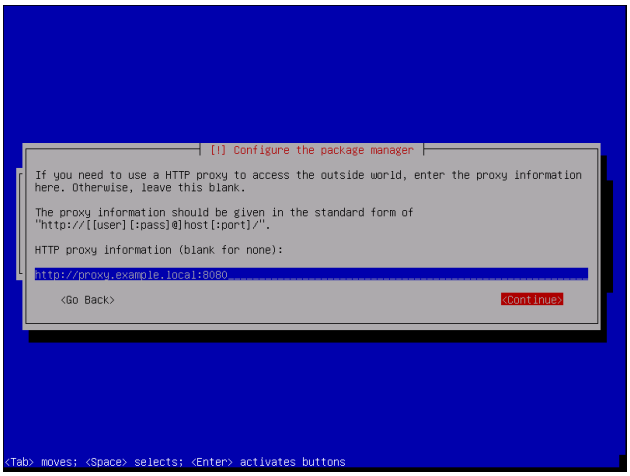

[?] Configure the package manager
Please select the protocol to be used for downloading files. If unsure, select "http"; it is less prone to problems involving firewalls.
Protocol for file downloads:
http
ftp
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons


[!] Configure the package manager
The goal is to find a mirror of the Debian archive that is close to you on the network -- be aware that nearby countries, or even your own, may not be the best choice.
Debian archive mirror country:
Australia
Austria
Bangladesh
Belarus
Belgium
Bosnia and Herzegovina
Brazil
Bulgaria
Canada
Chile
China
Colombia
Costa Rica
Croatia
Czech Republic
Denmark
El Salvador
Estonia
Finland
France
French Polynesia
Georgia
Germany
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons

Select the mirror server



Enter HTTP proxy information (leave blank for none)


[!] Configure the package manager
If you need to use a HTTP proxy to access the outside world, enter the proxy information
here. Otherwise, leave this blank.
The proxy information should be given in the standard form of
"http://[[user][:pass]@]host[:port]/".
HTTP proxy information (blank for none):
http://proxy.example.local:8080
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

Configuring apt

Select and install software

7. Completion

   After completing the installation the system will boot automatically. The system is based on Debian Wheezy kernel 3.2

   Now you can see the login-prompt with a link where you can access openATTIC via the browser. Please login as user "`openattic`" and password "`openattic`".

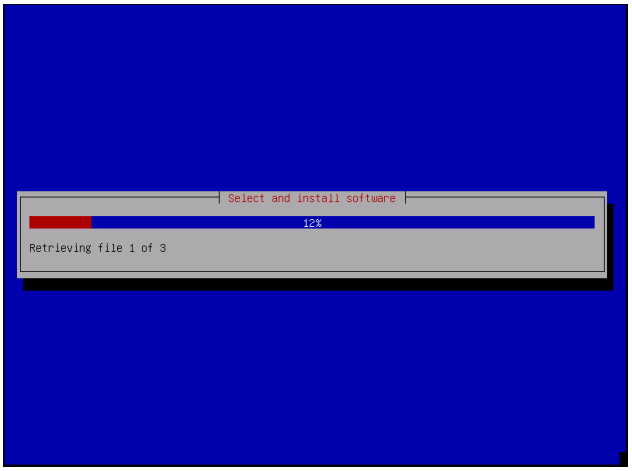
4.3.3 Installation in Univention Corporate Server(UCS)

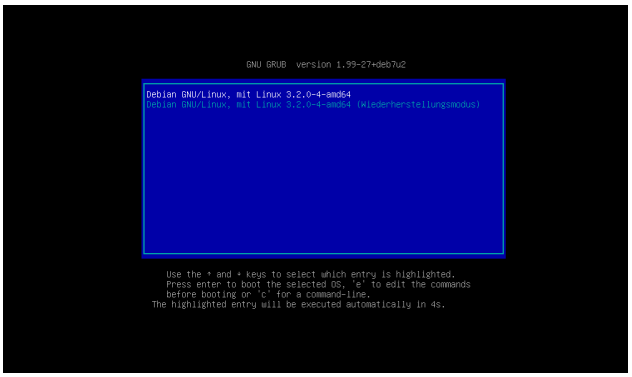
Since version 1.0 openATTIC supports also the installation under Univention Corporate Server 3.1.

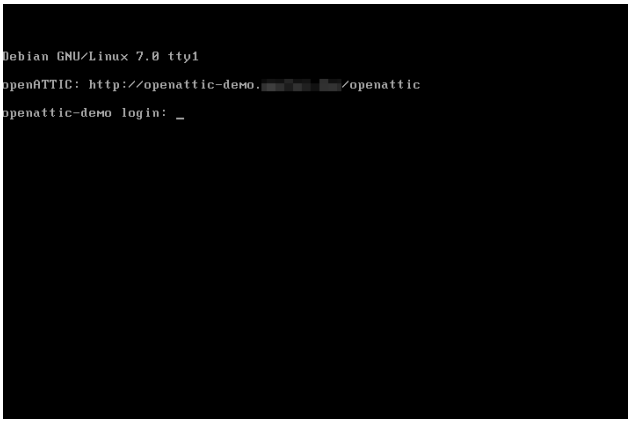
Requirements

- UCS-Member-Server

Configuring apt
43%
Retrieving file 9 of 9
<Cancel>


Select and install software
12%
Retrieving file 1 of 3


GNU GRUB version 1.99-27+deb7u2
Debian GNU/Linux, mit Linux 3.2.0-4-amd64
Debian GNU/Linux, mit Linux 3.2.0-4-amd64 (Wiederherstellungsmodus)
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line.
The highlighted entry will be executed automatically in 4s.



- The system mustn't be configured as nagios server
- `amd64` architecture
- At least 2GB RAM
- Free space in the configured UCS volume group or a free disk

Installation

1. **Insert the following repositories into `/etc/apt/sources.list`:**

   deb http://apt.open-attic.org/ ucs3.1-1 main

   deb-src http://apt.open-attic.org/ ucs3.1-1 main

2. Install the package `openattic-ucs`. There is a possibility that some configuration parameters will be promted in the course of the installation. Further informations can be found here: <REFERENZ KONFIRUGRATION DER PAKETE>.
3. Run `oaconfig install`.
4. **The next step depends on whether the system is a domain member server or not.**
   - If not, please add the server to the domain
   - If yes, please go to the menupoint "join domain" within the UCS gui. There is now another join script named `40openattic`. Run this join script.
5. While installation openATTIC has been integrated into the UCS overview page and is now ready for access.
1. You can now acceess openATTIC via you domain user.

The installation is now complete.

4.4 Package-based installation on Debian and Ubuntu

In order to use the Apt repository of openATTIC, create a file under `/etc/apt/sources.list/` named `openattic.list`.

Insert the following lines:

- Debian:

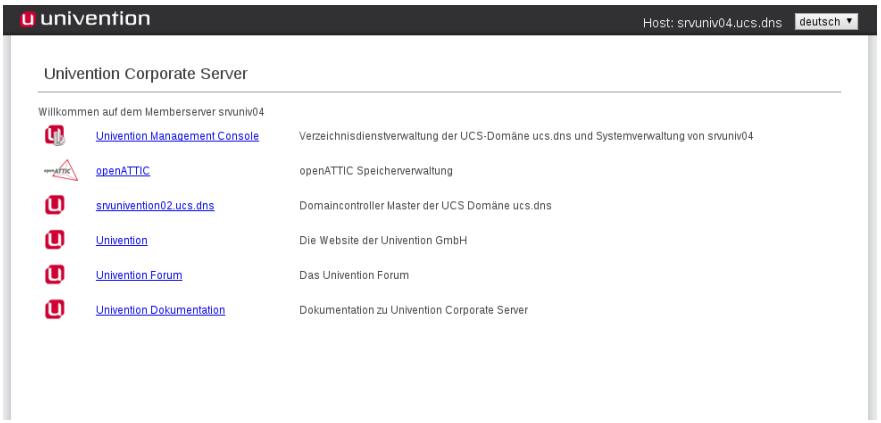


Figure 4.1: Univention Corporate Server overview

```
deb     http://apt.open-attic.org/ wheezy   main
deb-src http://apt.open-attic.org/ wheezy   main
```

- Ubuntu:

```
deb     http://apt.open-attic.org/ trusty  main
deb-src http://apt.open-attic.org/ trusty  main
```

Note: Please make sure the `linux-modules-extra` package for your kernel version is installed.

- Nightly build:

```
deb     http://apt.open-attic.org/ nightly  main
deb-src http://apt.open-attic.org/ nightly  main
```

We only support the amd64 architecture.

Importing the PGP key can be done with the following command:

```
apt-key adv --recv --keyserver hkp://keyserver.ubuntu.com A7D3EAFA
```

You can then proceed to install openATTIC by running:

- `apt-get update`
- `apt-get install openattic`
- `oaconfig install`

4.5 Installing an openATTIC cluster

openATTIC can be installed as a cluster, where any node can be used to manage the whole system and commands are distributed automatically.

4.5.1 Step 1 - Install two openATTIC hosts

In order to use DRBD®, we will need a cluster of two hosts. Install two openATTIC hosts as described in *Install*.

Note: **Important:** You should only execute the command `oaconfig install` on **one** of the two hosts, in the following example the command was executed on the host named *Alice*. This will result in the installation of the entire openATTIC system including the database.

In the following example the first host is called *Alice* (ip address: 172.16.14.41) and the second *Bob* (ip address: 172.16.14.42).

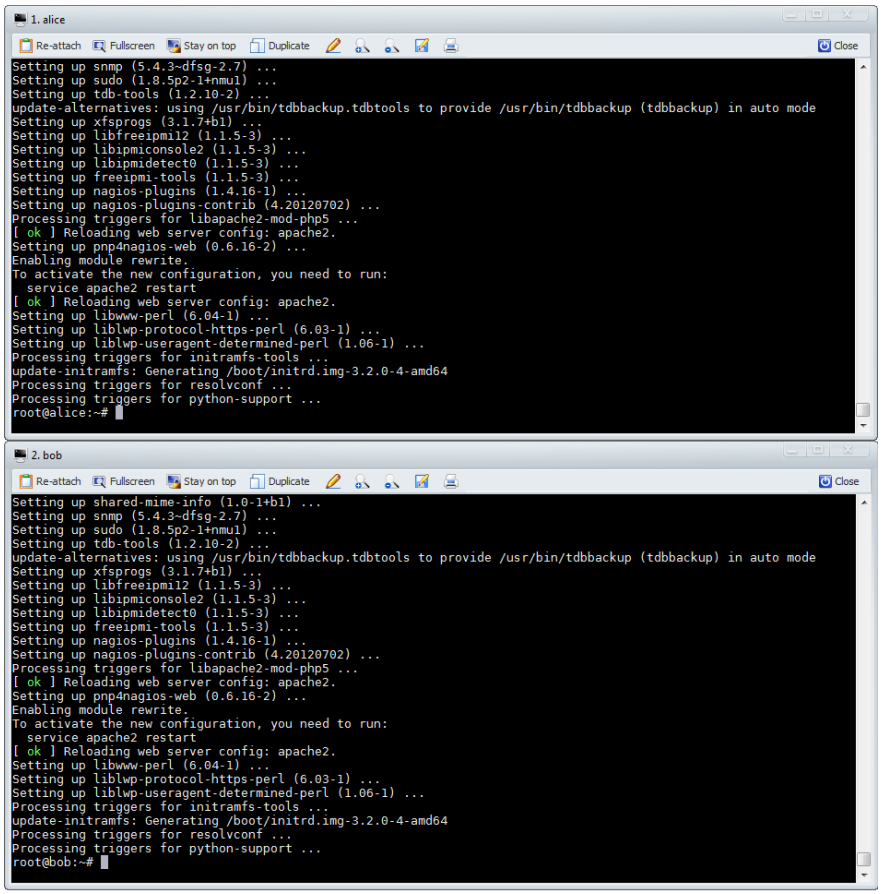

1. alice
Re-attach
Fullscreen
Stay on top
Duplicate
Close
Setting up snmp (5.4.3~dfsg-2.7) ...
Setting up sudo (1.8.5p2-1+nmu1) ...
Setting up tdb-tools (1.2.10-2) ...
update-alternatives: using /usr/bin/tdbbackup.tdbtools to provide /usr/bin/tdbbackup (tdbbackup) in auto mode
Setting up xfsprogs (3.1.7+b1) ...
Setting up libfreeipmi12 (1.1.5-3) ...
Setting up libipmiconsole2 (1.1.5-3) ...
Setting up libipmidetect0 (1.1.5-3) ...
Setting up freeipmi-tools (1.1.5-3) ...
Setting up nagios-plugins (1.4.16-1) ...
Setting up nagios-plugins-contrib (4.20120702) ...
Processing triggers for libapache2-mod-php5 ...
[ok] Reloading web server config: apache2.
Setting up pnp4nagios-web (0.6.16-2) ...
Enabling module rewrite.
To activate the new configuration, you need to run:
service apache2 restart
[ok] Reloading web server config: apache2.
Setting up libwww-perl (6.04-1) ...
Setting up liblwp-protocol-https-perl (6.03-1) ...
Setting up liblwp-useragent-determined-perl (1.06-1) ...
Processing triggers for initramfs-tools ...
update-initramfs: Generating /boot/initrd.img-3.2.0-4-amd64
Processing triggers for resolvconf ...
Processing triggers for python-support ...
root@alice:~#
2. bob
Re-attach
Fullscreen
Stay on top
Duplicate
Close
Setting up shared-mime-info (1.0-1+b1) ...
Setting up snmp (5.4.3~dfsg-2.7) ...
Setting up sudo (1.8.5p2-1+nmu1) ...
Setting up tdb-tools (1.2.10-2) ...
update-alternatives: using /usr/bin/tdbbackup.tdbtools to provide /usr/bin/tdbbackup (tdbbackup) in auto mode
Setting up xfsprogs (3.1.7+b1) ...
Setting up libfreeipmi12 (1.1.5-3) ...
Setting up libipmiconsole2 (1.1.5-3) ...
Setting up libipmidetect0 (1.1.5-3) ...
Setting up freeipmi-tools (1.1.5-3) ...
Setting up nagios-plugins (1.4.16-1) ...
Setting up nagios-plugins-contrib (4.20120702) ...
Processing triggers for libapache2-mod-php5 ...
[ok] Reloading web server config: apache2.
Setting up pnp4nagios-web (0.6.16-2) ...
Enabling module rewrite.
To activate the new configuration, you need to run:
service apache2 restart
[ok] Reloading web server config: apache2.
Setting up libwww-perl (6.04-1) ...
Setting up liblwp-protocol-https-perl (6.03-1) ...
Setting up liblwp-useragent-determined-perl (1.06-1) ...
Processing triggers for initramfs-tools ...
update-initramfs: Generating /boot/initrd.img-3.2.0-4-amd64
Processing triggers for resolvconf ...
Processing triggers for python-support ...
root@bob:~#

Figure 4.2: Installing two openATTIC hosts

4.5.2 Step 2 - Database configuration on *Bob*

Since *Alice* needs to share her database with *Bob* you will have to enter the database information (database name, user, password and host) from *Alice* into the database configuration file `/etc/openattic/database.ini` manually. It should look something like this:

4.5.3 Step 3 - Database configuration on *Alice*

Now it's time to edit the `/etc/postgresql/<VERSION>/main/postgresql.conf` and `/etc/postgresql/<VERSION>/main/pg_hba.conf` configuration files on *Alice*.

Set the correct listen addresses within the `postgres.conf` file. Use *Bob*'s ip address as in the example:

Add *Alice*'s ip address to the `pg_hba.conf` file within the IPv4 local connections section as follows:

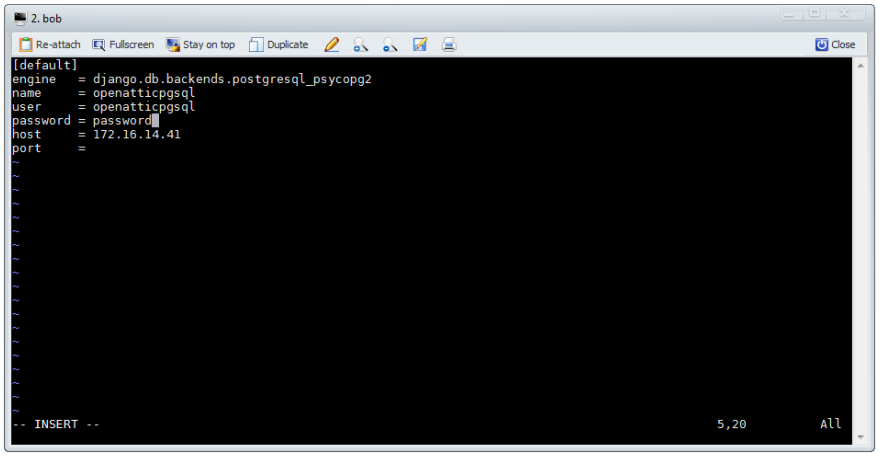


Figure 4.3: Database configuration on *Bob*

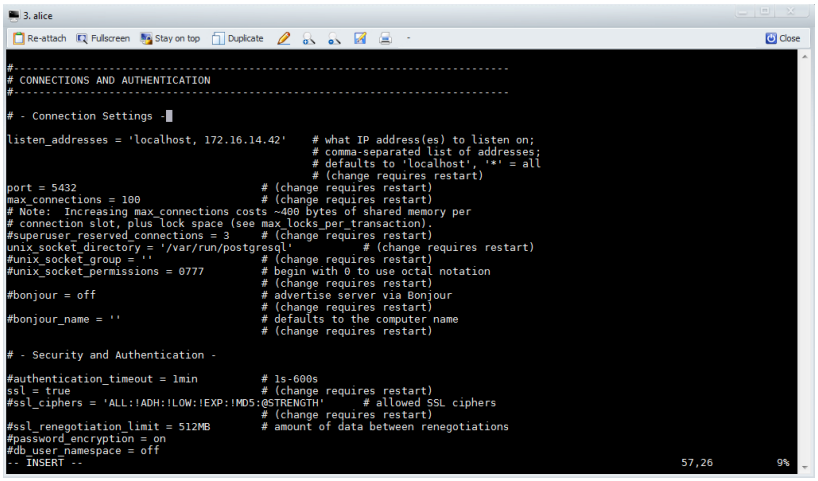

3. alice
Re-attach
Fullscreen
Stay on top
Duplicate
Close
# CONNECTIONS AND AUTHENTICATION
# - Connection Settings -
listen_addresses = 'localhost, 172.16.14.42' # what IP address(es) to listen on;
# comma-separated list of addresses;
# defaults to 'localhost', '*' = all
# (change requires restart)
port = 5432 # (change requires restart)
max_connections = 100 # (change requires restart)
# Note: Increasing max_connections costs ~400 bytes of shared memory per
# connection slot, plus lock space (see max_locks_per_transaction).
#superuser_reserved_connections = 3 # (change requires restart)
unix_socket_directory = '/var/run/postgresql' # (change requires restart)
#unix_socket_group = '' # (change requires restart)
#unix_socket_permissions = 0777 # begin with 0 to use octal notation
(change requires restart)
#bonjour = off # advertise server via Bonjour
(change requires restart)
#bonjour_name = '' # defaults to the computer name
(change requires restart)
- Security and Authentication -
#authentication_timeout = 1min # 1s-600s
ssl = true # (change requires restart)
#ssl_ciphers = 'ALL:!ADH:!LOW:!EXP:!MD5:@STRENGTH' # allowed SSL ciphers
(change requires restart)
#ssl_renegotiation_limit = 512MB # amount of data between renegotiations
#password_encryption = on
#db_user_namespace = off
-- INSERT --
57,26
9%

Figure 4.4: Configuration file `postgres.conf` on *Alice*

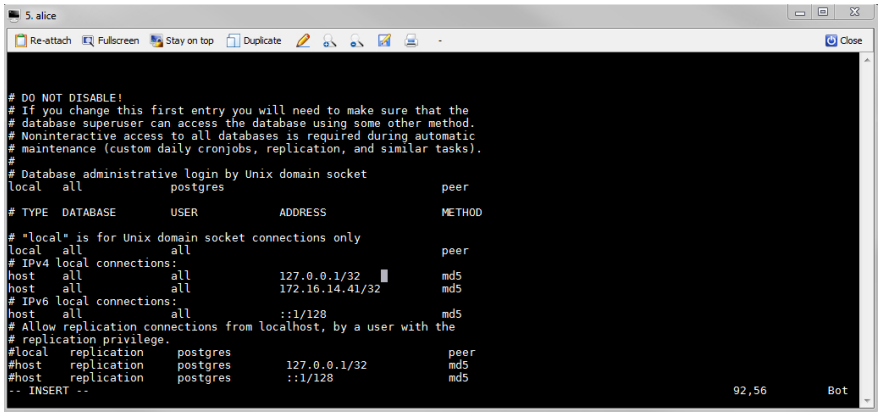

5. alice
Re-attach
Fullscreen
Stay on top
Duplicate
Close
DO NOT DISABLE!
If you change this first entry you will need to make sure that the
database superuser can access the database using some other method.
Noninteractive access to all databases is required during automatic
maintenance (custom daily cronjobs, replication, and similar tasks).
#
Database administrative login by Unix domain socket
local all postgres peer
TYPE DATABASE USER ADDRESS METHOD
"local" is for Unix domain socket connections only
local all all peer
IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all 172.16.14.41/32 md5
IPv6 local connections:
host all all ::1/128 md5
Allow replication connections from localhost, by a user with the
replication privilege.
#local replication postgres peer
#host replication postgres 127.0.0.1/32 md5
#host replication postgres ::1/128 md5
-- INSERT --
92,56
Bot

Figure 4.5: Configuration file `pg_hba.conf` on *Alice*

Now you can restart the postgresql service.

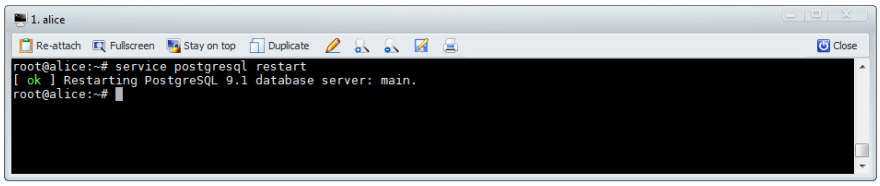


Figure 4.6: Restart the postgresql service on *Alice*

4.5.4 Step 4 - Execute `oaconfig install` on *Bob*

Now that you have hooked up *Bob* with *Alice*'s database you can install openATTIC on *Bob* by executing `oaconfig install`.

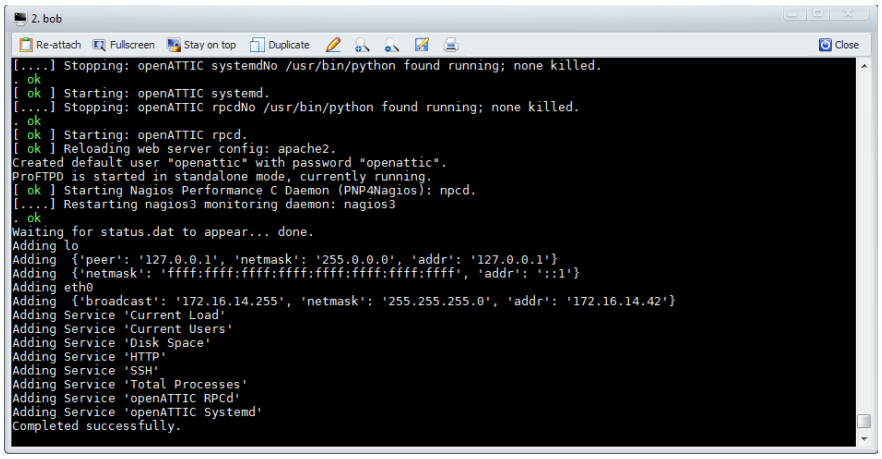

2. bob
Re-attach
Fullscreen
Stay on top
Duplicate
Close
[....] Stopping: openATTIC systemdNo /usr/bin/python found running; none killed.
. ok
[ok] Starting: openATTIC systemd.
[....] Stopping: openATTIC rpcdNo /usr/bin/python found running; none killed.
. ok
[ok] Starting: openATTIC rpcd.
[ok] Reloading web server config: apache2.
Created default user "openattic" with password "openattic".
ProFTPD is started in standalone mode, currently running.
[ok] Starting Nagios Performance C Daemon (PNP4Nagios): npcd.
[....] Restarting nagios3 monitoring daemon: nagios3
. ok
Waiting for status.dat to appear... done.
Adding lo
Adding {'peer': '127.0.0.1', 'netmask': '255.0.0.0', 'addr': '127.0.0.1'}
Adding {'netmask': 'ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff', 'addr': '::1'}
Adding eth0
Adding {'broadcast': '172.16.14.255', 'netmask': '255.255.255.0', 'addr': '172.16.14.42'}
Adding Service 'Current Load'
Adding Service 'Current Users'
Adding Service 'Disk Space'
Adding Service 'HTTP'
Adding Service 'SSH'
Adding Service 'Total Processes'
Adding Service 'openATTIC RPCd'
Adding Service 'openATTIC Systemd'
Completed successfully.

Figure 4.7: `oaconfig install` on *Bob*

4.6 Joining openATTIC to a Windows domain

Joining a domain is super easy using this command. Keep your domain name and administrator password handy:

```
root@debpkgtest:~# oaconfig domainjoin mziegler master.dns SAPDOM
User:            mziegler
Domain:          master.dns
Realm:           MASTER.DNS
Workgroup:       SAPDOM
Machine Account: DEBPKGTEST$
Updating krb5.conf...
Probing Kerberos...
Password for mziegler@MASTER.DNS: ********
Configuring Samba...
method return sender=:1.248 -> dest=:1.251 reply_serial=2
Removing old keytab...
Joining Domain...
Enter mziegler's password: ********
```

```
Using short domain name -- SAPDOM
Joined 'DEBPKGTEST' to realm 'master.dns'
Processing principals to add...
Logging in as DEBPKGTEST$ (this may fail a couple of times)...
kinit: Preauthentication failed while getting initial credentials
kinit: Preauthentication failed while getting initial credentials
Configuring openATTIC...
[ ok ] Stopping: openATTIC systemd.
[ ok ] Starting: openATTIC systemd.
[ ok ] Stopping: openATTIC rpcd.
[ ok ] Starting: openATTIC rpcd.
[ ok ] Reloading web server config: apache2.
Configuring libnss...
Restarting Samba and Winbind...
Initialized config from /etc/openattic/cli.conf
Could not connect to the server: [Errno 111] Connection refused
Initialized config from /etc/openattic/cli.conf
pong
method return sender=:1.252 -> dest=:1.253 reply_serial=2
[ ok ] Stopping Samba daemons: nmbd smbd.
[ ok ] Starting Samba daemons: nmbd smbd.
[ ok ] Stopping the Winbind daemon: winbind.
[ ok ] Starting the Winbind daemon: winbind.
To see if it worked, let's try 'getent passwd "mziegler"':
mziegler:*:20422:10513:Ziegler, Michael:/home/SAPDOM/mziegler:/bin/true
```

4.7 Configuring Authentication and Single Sign-On

When logging in, each user passes through two phases: Authentication and Authorization. The authentication phase employs mechanisms to ensure the users are who they say they are. The authorization phase then checks if that user is allowed access.

4.7.1 Authentication

openATTIC supports three authentication providers:

1. Its internal database. If a user is known to the database and they entered their password correctly, authentication is passed.
2. Using Pluggable Authentication Modules to delegate authentication of username and password to the Linux operating system. If PAM accepts the credentials, a database user without any permissions is created and authentication is passed.
3. Using Kerberos tickets via mod_auth_kerb. Apache will verify the Kerberos ticket and tell openATTIC the username the ticket is valid for, if any. openATTIC will then create a database user without any permissions and pass authentication.

4.7.2 Authorization

Once users have been authenticated, the authorization phase makes sure that users are only granted access to the openATTIC GUI if they posess the necessary permissions.

Authorization is always checked against the openATTIC user database. In order to pass authorization, a user account must be marked active and a staff member.

Users created by the PAM and Kerberos authentication backends will automatically be marked active, but will not be staff members. Otherwise, *every* user in your domain would automatically gain access to openATTIC, which is usually not desired.

However, usually there is a distinct group of users which are designated openATTIC administrators and therefore should be allowed to access all openATTIC systems, without needing to be configured on every single one.

In order to achieve that, openATTIC allows the name of a system group to be configured. During the authorization phase, if a user is active but not a staff member, openATTIC will then check if the user is a member of the configured user group, and if so, make them a staff member automatically.

4.7.3 Configuring Domain authentication and Single Sign-On

To configure authentication via a domain and to use Single Sign-On via Kerberos, a few steps are required.

1. Configuring openATTIC

   As part of the domain join process, the `oaconfig` script creates a file named `/etc/openattic/domain.ini` which contains all the relevant settings in Python's ConfigParser format.

   The `[domain]` section contains the kerberos `realm` and Windows `workgroup` name.

   The `[pam]` section allows you to enable password-based domain account authentication, and allows you to change the name of the PAM service to be queried using the `service` parameter. Note that by default, the PAM backend changes user names to upper case before passing them on to PAM – change the `is_kerberos` parameter to `no` if this is not desired.

   Likewise, the `[kerberos]` section allows you to enable ticket-based domain account authentication.

   In order to make use of the domain group membership check, add a section named `[authz]` and set the `group` parameter to the name of your group in lower case, like so:

   ```
   [authz]
   group = io-oa
   ```

   To verify the group name, you can try the following on the shell:

   ```
   $ getent group io-oa
   io-oa:x:30174:s.rieger,lpaduano,dbreitung,kwagner,mziegler,jkuhn,tdehler
   ```

2. Configuring Apache

   Please take a look at `/etc/apache2/conf.d/openattic`. At the bottom, this file contains a configuration section for Kerberos. Uncomment the section, and adapt the settings to your domain.

   In order to activate the new configuration, run:

   ```
   apt-get install libapache2-mod-auth-kerb
   a2enmod auth_kerb
   a2enmod authnz_ldap
   service apache2 restart
   ```

3. Logging in with Internet Explorer should work already. Firefox requires you to configure the name of the domain in `about:config` under `network.negotiate-auth.trusted-uris`.

4.7.4 Troubleshooting

As this is Kerberos and LDAP we're talking about, you *will* run into trouble.

First of all, remember that the *Operating System* chapter *did* tell you to install NTP and make sure that `hostname --fqdn` outputs something that makes sense, and please double-check that this works before proceeding.

Now, enjoy this little list of error messages. We've provided the usual meanings for your convenience:

- `Client not found in Kerberos database while getting initial credentials`

  The KDC doesn't know the service (i.e., your domain join failed).

- `Preauthentication failed while getting initial credentials`

  Wrong password or `/etc/krb5.keytab` is outdated (the latter should not happen because `oaconfig domainjoin` ensures that it is up to date).

- `Generic preauthentication failure while getting initial credentials`

  `/etc/krb5.keytab` is outdated. Update it using these commands:

  ```
  net ads keytab flush
  net ads keytab create
  net ads keytab add HTTP
  ```

- `gss_acquire_cred() failed: Unspecified GSS failure. Minor code may provide more information (, )`

  Apache is not allowed to read `/etc/krb5.keytab`, or wrong `KrbServiceName` in `/etc/apache2/conf.d/openattic`.

4.8 Additional openATTIC Modules

Installing the `openattic` metapackage will get you started with a pre-defined set of openATTIC modules that should be adequate for most situations. However, openATTIC allows you to choose a different set of modules in accordance with your needs. For instance, the LVM module is installed by default, but you can completely remove it if you only wish to use ZFS in your setup.

4.8.1 LVM

Handles the partitioning of physical disks into volumes using the Linux Logical Volume Manager. LVM supports enterprise level volume management of disks and disk subsystems by grouping disks into volume groups. The total capacity of volume groups can be allocated to logical volumes, which are accessed as regular block devices. LVM also supports snapshots, thereby allowing you to instantly create a copy of a volume, even while it is being accessed.

Installing

1. Install the openattic-module-lvm package:

   ```
   oaconfig install openattic-module-lvm
   ```

2. Add a volume group.

   - If a volume group already exists on your system, the installation process will recognize it and add it to openATTIC automatically.
   - Otherwise, you can create one on an empty disk using the `oaconfig add-disk` command, for example:

```
oaconfig add-disk /dev/sdb vgstorage
```

This command would create the `vgstorage` volume group and make it available to openATTIC.

Ideally, the device you use here should be a hardware or software RAID, so that you don't lose any data if a disk fails.

3. The volume group is now available and can be used.

4.8.2 MDRAID

Allows the combination of multiple disks to a single device that distributes its data across the member disks in a way specified by the user, allowing for load balancing, failure tolerance or both. Usually used in combination with the LVM module.

Installing

1. Install the openattic-module-mdraid package:

   ```
   oaconfig install openattic-module-mdraid
   ```

2. This module will periodically scan for RAID devices and update their status in the GUI.

4.8.3 TWRAID

Manages 3ware RAID controller devices that group multiple disks into a single device in order to provide failure tolerance and caching.

Installing

1. The openATTIC TWRAID module requires the `tw-cli` utility to be available. It can be retrieved from the HWRAID for Linux repository, so please add it to your `/etc/apt/sources.list` file.
2. Install the openattic-module-twraid package:

   ```
   oaconfig install openattic-module-twraid
   ```

3. This module will periodically scan for RAID devices and update their status in the GUI.

4.8.4 ZFS

Next-generation file system that handles volume-creation, snapshots, disk management, failure tolerance and caching. Its use is recommended for running large file servers on slow disks.

Installing

1. The openATTIC ZFS module requires zfsonlinux to be available, so please make sure you include the necessary sources for your distribution in your `/etc/apt/sources.list` file.
2. Install the openattic-module-zfs package:

```
oaconfig install openattic-module-zfs
```

3. If you have LVM installed, you can now format newly created logical volumes using ZFS to create ZPools.
4. You can also create ZPools on the command line, and run `oaconfig install` (without any packages) make them available to openATTIC.

4.8.5 BTRFS

While ZFS was originally developed on Solaris and is now being ported over to Linux, BTRFS, which aims to provide the same functionality, was designed specifically for linux. This results in advanced features like full Windows ACL support, and better support for different kernel versions. If you want to use openATTIC as a FibreChannel target, using ZFS on the same node is going to be more challenging than using BTRFS.

However, BTRFS is still in development and is considered experimental.

Installing

Currently, the BTRFS module works best in combination with the LVM module.

1. Install the openattic-module-btrfs package:

   ```
   oaconfig install openattic-module-btrfs
   ```

2. You can now format newly-created logical volumes using BTRFS.

4.8.6 DRBD®

Supports mirroring of block devices to another host, thereby enabling failure tolerance.

Installing

1. The DRBD® module requires a working openATTIC cluster with a minimum of two nodes. Please refer to *Installing an openATTIC cluster* for further information on how to configure a cluster.
2. Once openATTIC has been installed on *Bob* it's time to install the openATTIC DRBD® module by executing `oaconfig install openattic-module-drbd` on both hosts.

Now that the hosts are set up correctly, we can create a volume in the openATTIC GUI to mirror it via DRBD® on a different host. The next step will be to create a volume in the openATTIC user interface and mirror it via DRBD®. Please have a look at the *mirror_conf* section to see how that happens.

4.8.7 Ceph

Allows the combination of storage on a large number of hosts into a single entity. Data placement and redundancy is taken care of automatically and scaling the system is simplified drastically. If you need storage without limits, Ceph is the way to go.

Note: We are currently working on this part of our documentation. An update will be available soon.

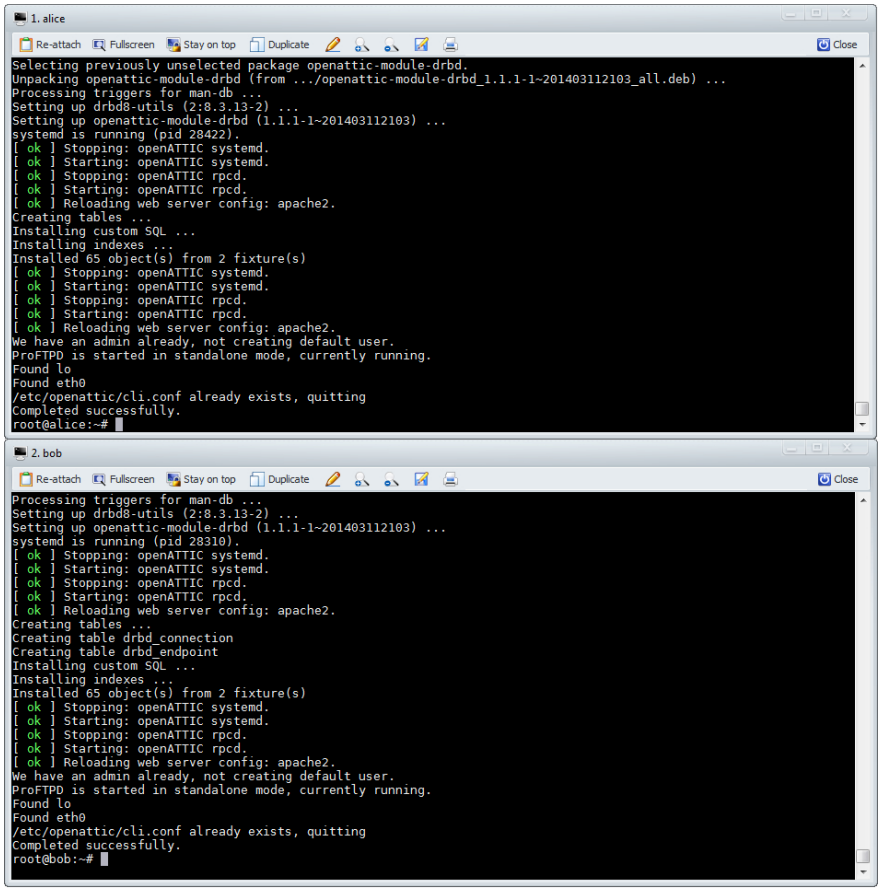

1. alice
Re-attach
Fullscreen
Stay on top
Duplicate
Close
Selecting previously unselected package openattic-module-drbd.
Unpacking openattic-module-drbd (from .../openattic-module-drbd_1.1.1-1~201403112103_all.deb) ...
Processing triggers for man-db ...
Setting up drbd8-utils (2:8.3.13-2) ...
Setting up openattic-module-drbd (1.1.1-1~201403112103) ...
systemd is running (pid 28422).
[ok] Stopping: openATTIC systemd.
[ok] Starting: openATTIC systemd.
[ok] Stopping: openATTIC rpcd.
[ok] Starting: openATTIC rpcd.
[ok] Reloading web server config: apache2.
Creating tables ...
Installing custom SQL ...
Installing indexes ...
Installed 65 object(s) from 2 fixture(s)
[ok] Stopping: openATTIC systemd.
[ok] Starting: openATTIC systemd.
[ok] Stopping: openATTIC rpcd.
[ok] Starting: openATTIC rpcd.
[ok] Reloading web server config: apache2.
We have an admin already, not creating default user.
ProFTPD is started in standalone mode, currently running.
Found lo
Found eth0
/etc/openattic/cli.conf already exists, quitting
Completed successfully.
root@alice:~#
2. bob
Re-attach
Fullscreen
Stay on top
Duplicate
Close
Processing triggers for man-db ...
Setting up drbd8-utils (2:8.3.13-2) ...
Setting up openattic-module-drbd (1.1.1-1~201403112103) ...
systemd is running (pid 28310).
[ok] Stopping: openATTIC systemd.
[ok] Starting: openATTIC systemd.
[ok] Stopping: openATTIC rpcd.
[ok] Starting: openATTIC rpcd.
[ok] Reloading web server config: apache2.
Creating tables ...
Creating table drbd_connection
Creating table drbd_endpoint
Installing custom SQL ...
Installing indexes ...
Installed 65 object(s) from 2 fixture(s)
[ok] Stopping: openATTIC systemd.
[ok] Starting: openATTIC systemd.
[ok] Stopping: openATTIC rpcd.
[ok] Starting: openATTIC rpcd.
[ok] Reloading web server config: apache2.
We have an admin already, not creating default user.
ProFTPD is started in standalone mode, currently running.
Found lo
Found eth0
/etc/openattic/cli.conf already exists, quitting
Completed successfully.
root@bob:~#

Figure 4.8: Installing the DRBD® module

4.8.8 IPMI

Queries hardware sensors for better hardware monitoring and failure detection.

Installing

1. Install the openattic-module-ipmi package:

   ```
   oaconfig install openattic-module-ipmi
   ```

2. In the GUI, there is now a menu entry named Sensors that displays the status of your hardware.

4.9 Maintenance

4.9.1 Commandline-Tool `oaconfig`

Although openATTICs' graphical interface is the most important interface for the user, it is necessary to execute actions via the commandline. Therefore you can use either the tool *oacli* , which gives you the possibility to access the RPC-API via the commandline and offers all options/actions which are also supported from the graphical user interface or the script *oaconfig* for system-based operations, which takes over some administrative tasks and will be annotated in the following section.

If you use the command *oaconfig* you need to pass an argument. In case you do not pass any commands or use `oaconfig help` a list of all possible commands will be listed. Some of the listed commands are implemented by *oaconfig* itself, others from the administrative subsystem of openATTIC. Which commands are listed, depends on the openATTIC installation.

To get help for a sub-command, execute `oaconfig <command> --help`.

basic commands

The following commands are always available:

- `install`: Initializes the installed modules. It needs to be executed after installing the base packages or after installing *additional modules*.
- `install-cli`: creates an API-Key and a config file for oacli.
- `scan-vgs`: instructs LVM to search for existing physical volumes and volume groups. They will be searched but not recorded into the database.
- `add-vg`: Adds an already existing volume group into the database.
- `add-disk`: formats a disk and adds it to a volume group. In case the volume group does not exist yet, it will be created and added to the database.

  > **Warning:** This command skips the validation of the graphical interface if the disk is already in use. Only standard validation of creating a physical volume from `pvcreate` will be executed.

  Please be careful if you want to use this command.
- `restart`: Restarts all openATTIC services.
- `reload`: Reloads the configs of all openATTIC services.
- `status`: Shows the current states of the openATTIC services.

- `rootshell`: Starts a python-shell as root user.

Administrative-Subsystem

Important commands of the administrative subsystems are:

- `syncdb`: Creates a database schema. It is recommended to **not** use this command itself use the command `oaconfig install` instead.
- `haveadmin`: Shows if at least one administrative user of the database exists.
- `mkapikey`: Creates an API-Key for a user.
- `changepassword`: Changes the password of a user.
- `shell`: Executes a python shell as openattic-user.

4.9.2 How to backup the openATTIC database

CHAPTER

FIVE

IMPLEMENTING THE USE CASES

As outlined in *Use Cases*, there are a number of ways of setting up openATTIC. Once the *initial installation* has been completed, you can proceed to integrating openATTIC into your environment, depending on your needs. The guides in this chapter provide step-by-step tutorials on how to build the various setup variants.

5.1 Implementing a file server

This section describes the implementation procedure for the *Fileserver* use case. Please refer to this section for more general considerations.

5.1.1 Prerequisites

1. File servers usually don't impose latency-critical requirements upon the storage system. The limiting factors are storage capacity and bandwidth. This means that using bigger, slower disks is fine.

   However, care needs to be taken when it comes to parity: The bigger your disks are, the more likely a rebuild will fail due to an unrecoverable read error (URE). Make sure you either have at least double parity (using RAID6 or raidz2), or your disks have a low URE rate.

2. Make sure user authentication works by *joining openATTIC into your Windows domain*.

5.1.2 Basic setup

1. Create a volume.
   (a) Make sure the volume you create fulfills your capacity requirements.
   (b) The volume pool you use should reside on bigger, slower disks if you have different disk types to choose from.
   (c) We recommend using Ext4 or btrfs as the file system.

   Note: XFS's performance advantages are only relevant for latency-critical applications, which a file server is not.

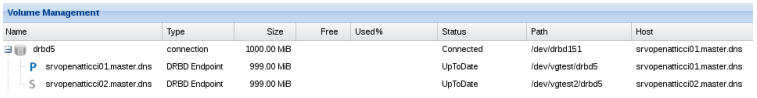
   Note: ZFS does not support Windows ACLs, so the enforcement of permissions is limited when using ZFS.

2. *Create a CIFS share* to export the volume.
3. Connect to the share using Windows Explorer and configure permissions.

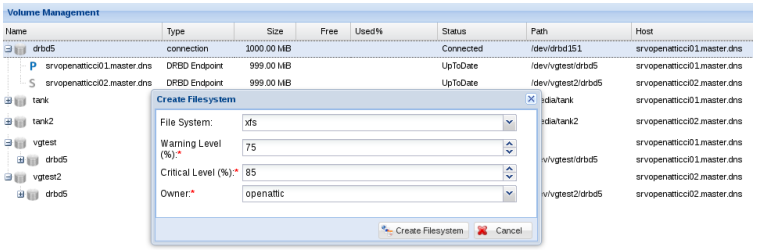
5.1.3 Clustered setup

1. Create a *mirrored volume*.

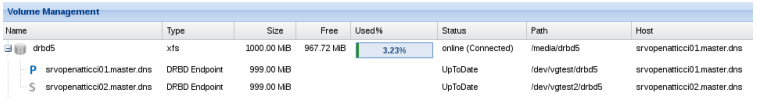
   In the volume management, the volume should look like this:



2. Select the volume and click the "Create Filesystem" button. In the window that appears, choose Ext4 for the file system, choose the initial owner of the file system, and click the "Create Filesystem" button.



   The volume management should now display the DRBD connection with a file system and it should look like this:



Note: Creating the file system takes a little while, especially if your DRBD connection is still in the initial synchronization phase. Until the file system has been fully created, the volume will be in the `locked` state. Please wait a moment for the file system to be fully created.

---

3. *Create a CIFS share* to export the volume.
4. Connect to the share using Windows Explorer and configure permissions.

5.1.4 Snapshots

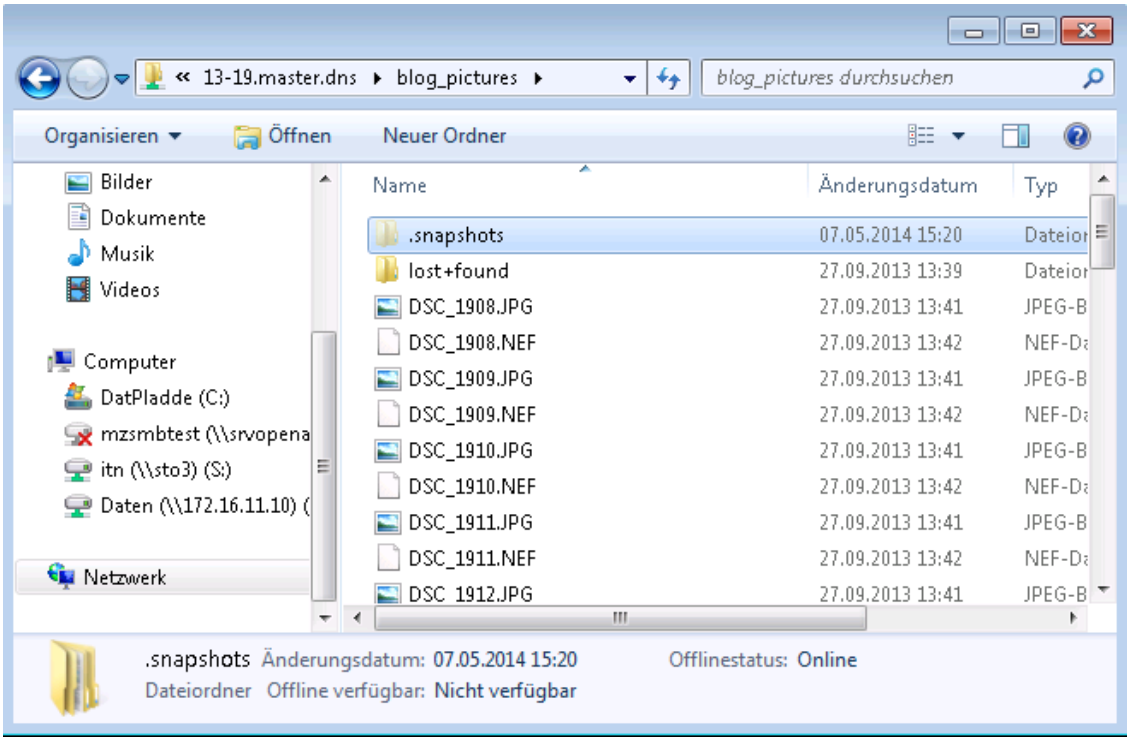
openATTIC supports creation and export of snapshots of the shared volume. Those will be visible in a hidden directory in the share's root directory called `.snapshots`, in which snapshots will be available.

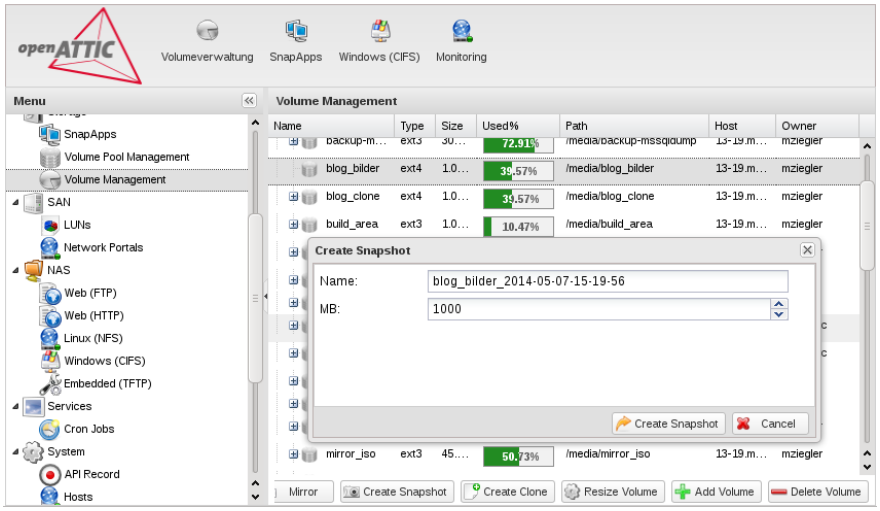
This directory will be populated automatically when you create a snapshot using the volume management:

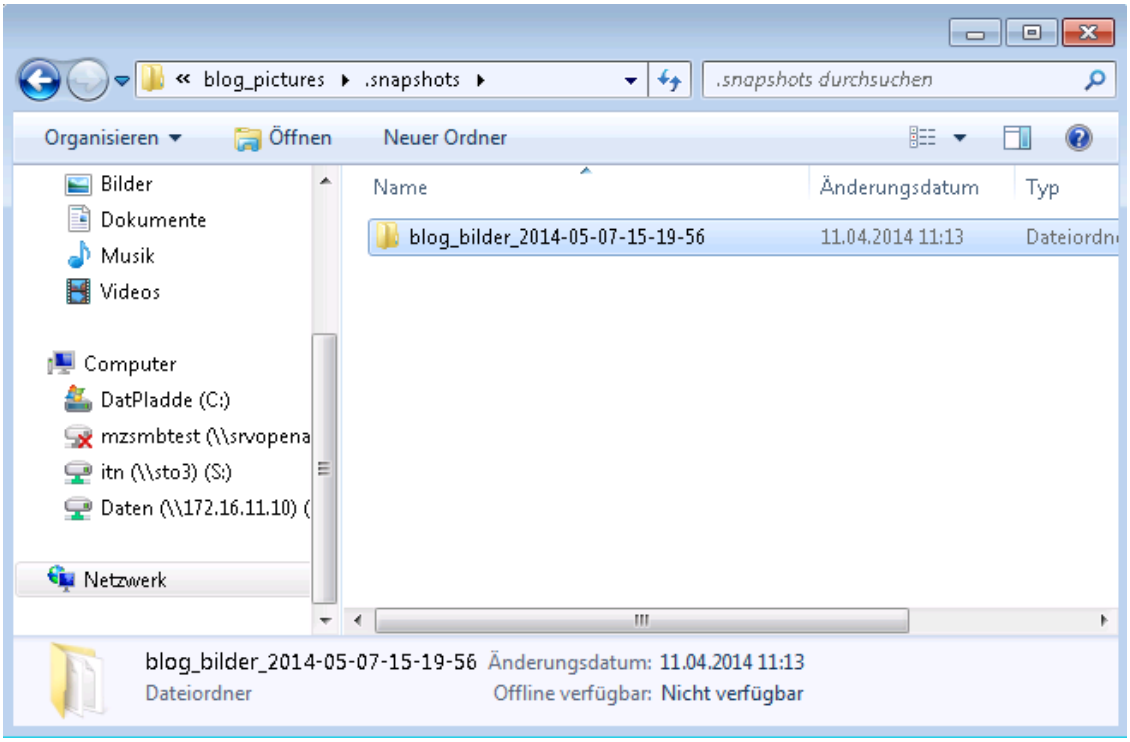
Each snapshot will be available as a subdirectory inside `.snapshots`:

5.1.5 Mirror servers

In order to set up a mirror server, follow the procedure outlined above and simply use an *HTTP export* instead of a CIFS share.


« 13-19.master.dns ▸ blog_pictures ▸
blog_pictures durchsuchen
Organisieren
Öffnen
Neuer Ordner
Bilder
Dokumente
Musik
Videos
Computer
DatPladde (C:)
mzsmbtest (\\srvopena
itn (\\sto3) (S:)
Daten (\\172.16.11.10) (
Netzwerk
Name
Änderungsdatum
Typ
.snapshots 07.05.2014 15:20
lost+found 27.09.2013 13:39
DSC_1908.JPG 27.09.2013 13:41
DSC_1908.NEF 27.09.2013 13:42
DSC_1909.JPG 27.09.2013 13:41
DSC_1909.NEF 27.09.2013 13:42
DSC_1910.JPG 27.09.2013 13:41
DSC_1910.NEF 27.09.2013 13:42
DSC_1911.JPG 27.09.2013 13:41
DSC_1911.NEF 27.09.2013 13:42
DSC_1912.JPG 27.09.2013 13:41
.snapshots Änderungsdatum: 07.05.2014 15:20
Dateiordner Offline verfügbar: Nicht verfügbar
Offlinestatus: Online


openATTIC
Volumeverwaltung
SnapApps
Windows (CIFS)
Monitoring
Menu
Volume Management
SnapApps
Volume Pool Management
Volume Management
SAN
LUNs
Network Portals
NAS
Web (FTP)
Web (HTTP)
Linux (NFS)
Windows (CIFS)
Embedded (TFTP)
Services
Cron Jobs
System
API Record
Hosts
Name
Type
Size
Used%
Path
Host
Owner
blog_bilder ext4 1.0… 39.57% /media/blog_bilder 13-19.m… mziegler
blog_clone ext4 1.0… 39.57% /media/blog_clone 13-19.m… mziegler
build_area ext3 1.0… 10.47% /media/build_area 13-19.m… mziegler
mirror_iso ext3 45.… 50.73% /media/mirror_iso 13-19.m… mziegler
Create Snapshot
Name: blog_bilder_2014-05-07-15-19-56
MB: 1000
Create Snapshot
Cancel
Mirror
Create Snapshot
Create Clone
Resize Volume
Add Volume
Delete Volume



Apt mirrors

Mirrors for the Apt package manager used by Debian and Ubuntu can be easily created using Debmirror.

RPM mirrors

Mirrors for RPM-based distributions (e.g. Fedora) are created and updated using rsync.

Software distribution for Windows clients

Windows clients can be managed using OPSI.

5.2 Implementing a virtualization storage

This section describes the implementation procedure for the *Virtualization* use case. Please refer to this section for more general considerations.

5.2.1 Prerequisites

1. Virtualization clusters have critical latency requirements, but do not normally require much bandwidth or suffer from fragmentation. Make sure that you run on smaller SAS disks, where single parity is sufficient and the disks are generally faster.

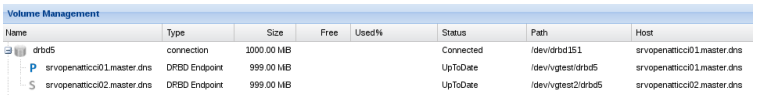
5.2.2 Basic setup

1. Create a *mirrored volume*.

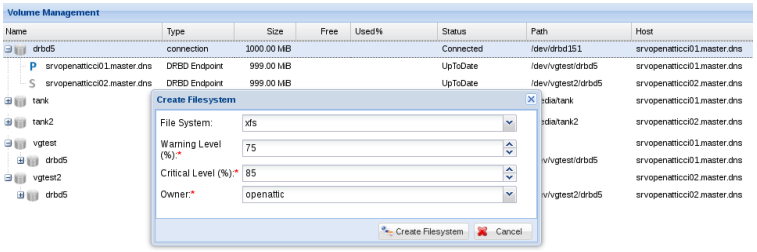
   (a) Make sure the volume you create fulfills your capacity requirements.

   (b) The volume pool you use should reside on smaller, faster disks if you have different disk types to choose from.

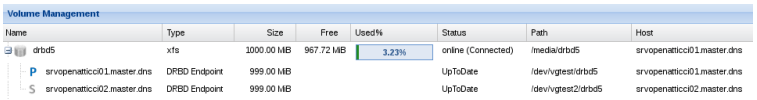
   In the volume management, the volume should look like this:



2. Select the volume and click the "Create Filesystem" button. In the window that appears, choose XFS for the file system, choose the initial owner of the file system, and click the "Create Filesystem" button.



   The volume management should now display the DRBD connection with a file system and it should look like this:



Note: Creating the file system takes a little while, especially if your DRBD connection is still in the initial synchronization phase. Until the file system has been fully created, the volume will be in the `locked` state. Please wait a moment for the file system to be fully created.

---

3. *Create an NFS share* to export the volume. Make sure that *all* nodes that need access to the share are accounted for, either by creating an export that includes the whole subnet, or by creating an export for each node.
4. Mount the share in your virtualization system as described in the following sections.

5.2.3 oVirt

To use the openATTIC volume for virtual machine deployment in oVirt, you have to add it as an NFS storage domain. To do so,

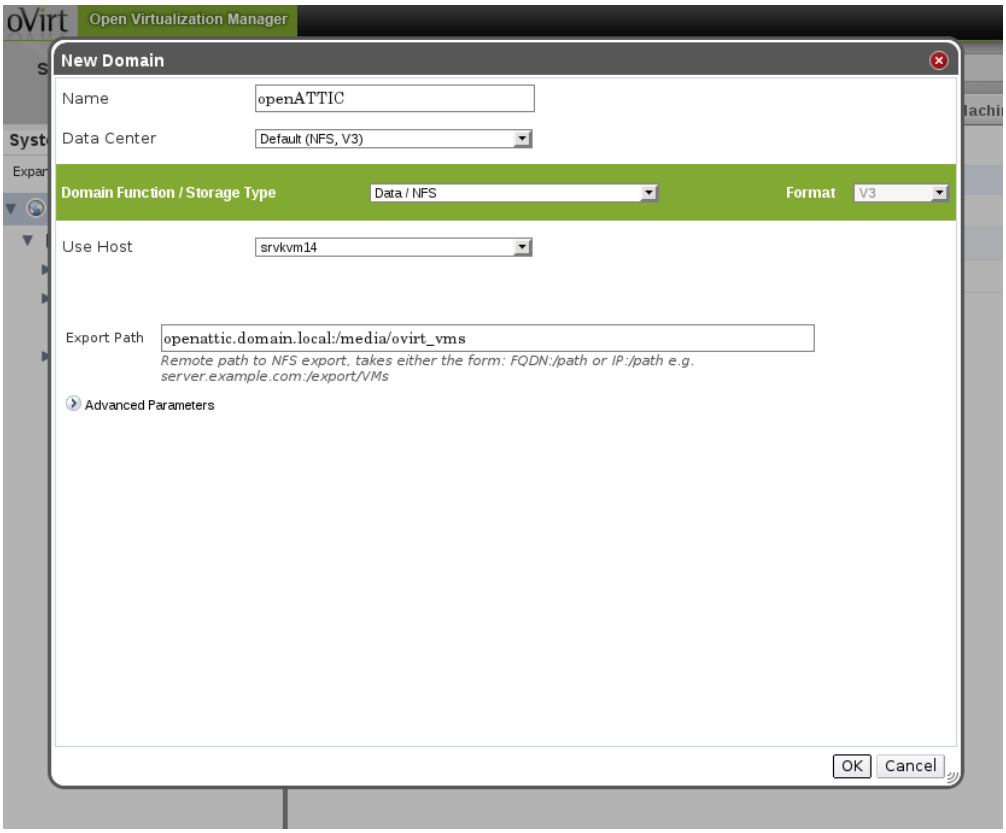
1. Log in to your openATTIC server using ssh, and make sure that the volume you wish to use for oVirt belongs to the user and group ID `36` by running:

   ```
   chown -R 36:36 /media/<volume>
   ```

   Otherwise, the storage domain may fail to initialize.

2. Your datacenter instance has to be created as an NFS type datacenter, which is the default.
3. Log in to your oVirt management system and select the "Storage" tab.
4. Click "New Domain".
5. Enter a name for the new storage domain.
6. Choose "Data / NFS" in the "domain function" field if you want VMs to be stored on this volume. Otherwise select the function you wish to use the volume for.
7. The export path has to contain both the address of the openATTIC server, as well as the path of the volume.

When done, it should look something like this:

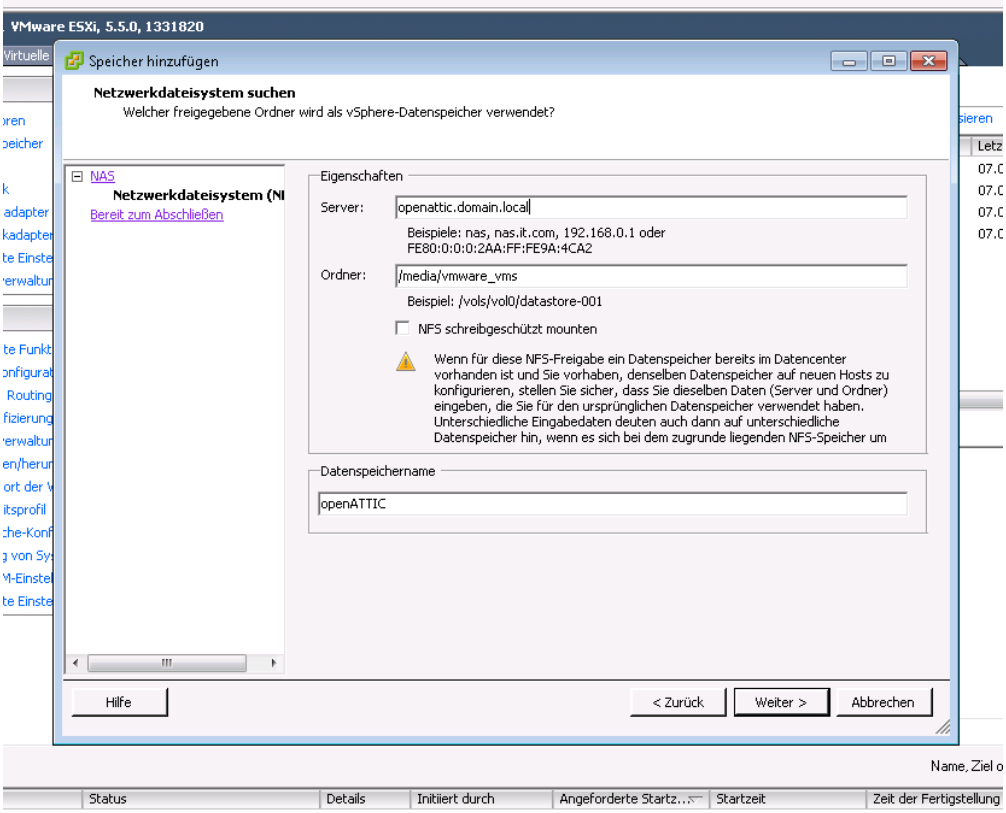


After you clicked the "OK" button, oVirt will start to initialize the volume and make it available as a storage domain.

5.2.4 VMware vSphere

Using VMware vSphere Client, log in to your vSphere Virtual Center and add the openATTIC volume as an NFS datastore to your virtualization hosts. To do so,

1. Click a host in the inventory screen and click on the "Configuration" tab.
2. In the hardware panel, choose "Storage" and click on "Add Storage".
3. Choose "Network File System (NFS)" as the storage type.
4. Enter the server's address, the volume path and a name for the datastore.


VMware ESXi, 5.5.0, 1331820
Speicher hinzufügen
Netzwerkdateisystem suchen
Welcher freigegebene Ordner wird als vSphere-Datenspeicher verwendet?
NAS
Netzwerkdateisystem (NI
Bereit zum Abschließen
Eigenschaften
Server:
openattic.domain.local
Beispiele: nas, nas.it.com, 192.168.0.1 oder FE80:0:0:0:2AA:FF:FE9A:4CA2
Ordner:
/media/vmware_vms
Beispiel: /vols/vol0/datastore-001
NFS schreibgeschützt mounten
Wenn für diese NFS-Freigabe ein Datenspeicher bereits im Datencenter vorhanden ist und Sie vorhaben, denselben Datenspeicher auf neuen Hosts zu konfigurieren, stellen Sie sicher, dass Sie dieselben Daten (Server und Ordner) eingeben, die Sie für den ursprünglichen Datenspeicher verwendet haben. Unterschiedliche Eingabedaten deuten auch dann auf unterschiedliche Datenspeicher hin, wenn es sich bei dem zugrunde liegenden NFS-Speicher um
Datenspeichername
openATTIC
Hilfe
< Zurück
Weiter >
Abbrechen
Name, Ziel o
Status
Details
Initiiert durch
Angeforderte Startz...
Startzeit
Zeit der Fertigstellung

When done, it should look something like this:

After you clicked the "Finish" button, vSphere will initialize the data store and make it available to be used by virtual machines.

Note: In order to support migration between multiple hosts, you have to add the datastore to *all* your hosts manually. Be sure to use *exactly* the same connection information on *all* your hosts, otherwise vSphere will not recognize the mounts belonging to the same datastore.

5.2.5 VirtualBox

VirtualBox does not care where virtual machines are actually stored, so simply mounting the NFS share on the VirtualBox host is sufficient. When mounting however, be sure to use NFS version 3:

```
mount -t nfs -o vers=3 openattic.domain.local:/media/virtualbox_vms /media/vms
```

Then configure VM images to reside in a directory underneath `/media/vms`.

5.3 Implementing Cloud Storage

This section describes the implementation procedure for the *Cloud Storage* use case. Please refer to this section for more general considerations.

5.3.1 IaaS: OpenStack, openQRM

How to make use of openATTIC storage for OpenStack depends on the actual project you're installing.

Glance

Note: We are currently working on this part of our documentation. An update will be available soon.

Cinder

The openATTIC team provides a Cinder driver.

See also:

OpenStack

Nova

Nova can be integrated in two ways:

1. Mount an NFS share to `/var/lib/nova/instances`.
2. Alternatively, you can configure Nova to use Ceph.

openQRM

The openATTIC team provides an openQRM plugin.

openQRM

5.3.2 Synchronized file storage: OwnCloud

1. Install ownCloud.
2. Create a volume to be used for ownCloud data.
3. Move the `/var/lib/owncloud/data` directory to your volume.
4. Create a symlink at `/var/lib/owncloud/data` that points to the volume.

CHAPTER

SIX

USER MANUAL

This section covers the openATTIC graphical user interface (GUI), focusing on storage tasks like adding volumes and shares, system management tasks like the configuration of users and API credentials, and the integrated monitoring system.

6.1 Status

6.1.1 Dashboard

You can configure the dashboard in order to create a personalized overview. For example to get an overview of the health state of your system, important disks or services. You don't have to waste time on searching for all the important components and their states in different panels, you can just summarize them here.

Add graphs by first selecting "Monitoring" from the menu tree on the left. Select the service and click the "Dashboard" button. You can change the suggested name for the graph if you want to and click "ok" and your graph will appear on the Dashboard.

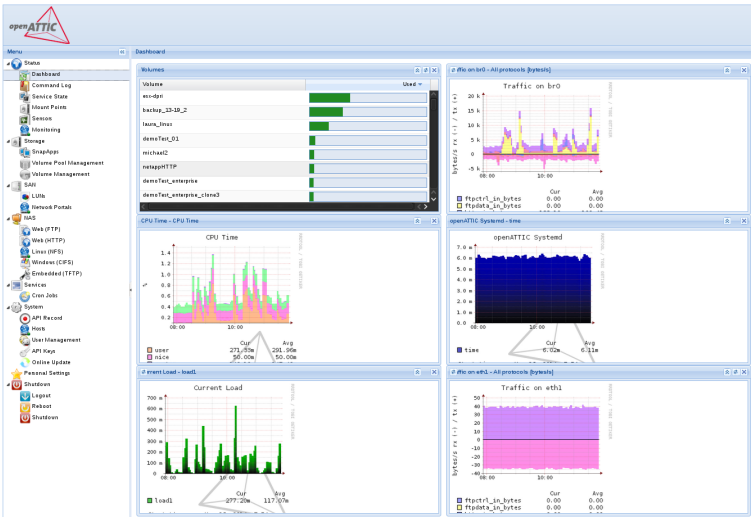


Figure 6.1: Dashboard panel

Remove graphs by clicking the "x" button in the right corner of the graph window in the dashboard overview.

If you want to reload the graph just click the the button in the middle in the right corner of the graph window. By clicking the left button in the right corner of the graph you can minimize the window. To maximize it, click the button again.

6.1.2 Command Log

The Command log lists commands executed in the background as result of actions taken in the user interface - for example the creation of a volume. When the user creates a volume the list of actions in the Command Log will include "/sbin/lvcreate". Furthermore, you can see if the execution of the command was successful or not.

Click the skip button to view old log entries.

It's also possible to **filter for commands with keywords** for example "nagios", "lvcreate" or for a specific date (format: mm/dd/yyyy, for example: 03/31/2014). Just click in the "Search" field and enter your keyword or a specific date - the panel will then display all matches.

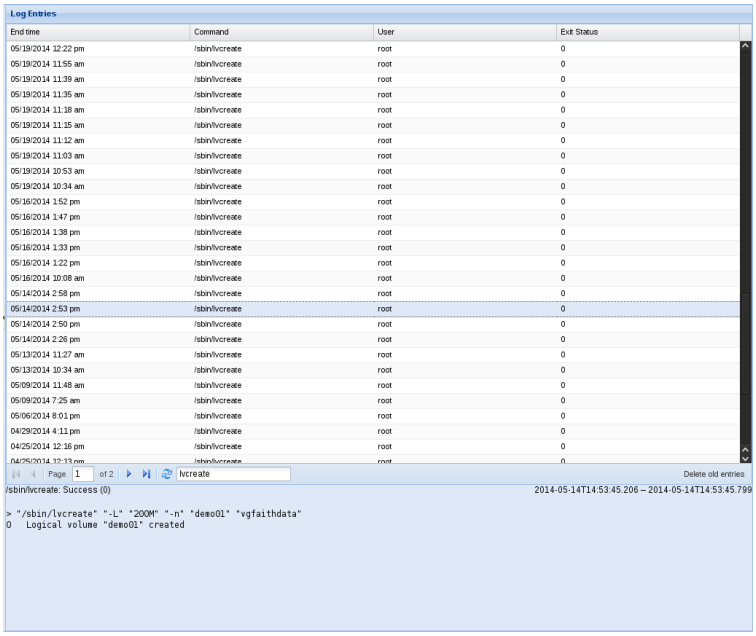
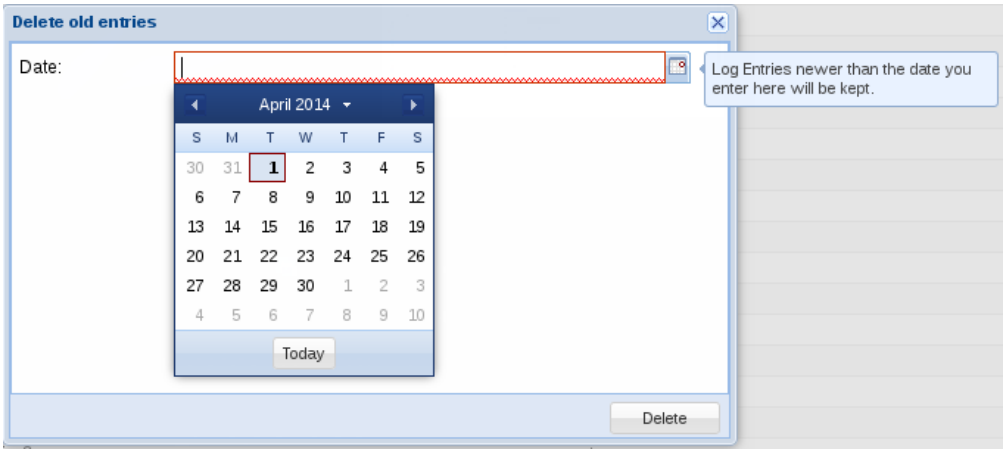


Figure 6.2: filter example

Delete old log entries by hitting the "Delete old entries" button on the right. You can enter the date manually or pick a date in the calendar.


Delete old entries
Date:
Log Entries newer than the date you enter here will be kept.
April 2014
S M T W T F S
30 31 1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 1 2 3
4 5 6 7 8 9 10
Today
Delete

6.1.3 Service State

Running openATTIC requires some services to be running as well – for example `openattic_rpcd`, `openattic_systemd` as well as `apache2` for web access. The Service state menu offers an overview of all important services and their current state.

You can start and stop services by selecting the service and clicking the button "Start" or "Stop".

Warning: Stopping important services like openattic_rpcd, openattic_systemd or apache2 (for a working webinterface) causes the openATTIC system to not work fluently.

Note: Please connect to the openATTIC system via ssh in case you're not able to restart the services via the webinterface. Type (i.e.) `/etc/init./openattic_rpcd start` or `/etc/init.d/apache2 start`. Normally, you will get a response which says that the service was started successfully.

Figure 6.3: Service State panel

6.1.4 Mount Points

The Mount Points panel offers an overview of all mount points and their properties:

- Device
- Mount Point
- File system Type of the device
- Options (attributes)

In the following text a more detailed description of the above listed points can be found.

openATTIC uses the package "mount" which is included in every standard installation of any Ubuntu/Debian version. When a user creates a volume the device will be mounted automatically in openATTIC. This is necessary in order to access the created volume and stored data on it - most common mount points are /dev, /mnt, /media.

To store data on a volume it needs to be formatted with a filesystem which will be done when the user selected a filesystem in the create volume process. When you left the filesystem field empty in order to map it to another host i.e. via iSCSI to a windows host you need to format the volume manually. In order to use the device, store data or open files from it, default options like "rw" (which means "read-write") will be passed as parameter when mounting a volume (device).

You can also find those parameters in the mount points panel under "Options". As mentioned before, there will be some default parameters passed when mounting a device - this depends on which filesystem the device will be formatted with and/or if the device is part of a raid.

Here is an example of options of a device with filesystem ext4:

```
rw              mount the device read-write

relatime        modified atime, reduces the fsync accesstime

user_xattr      extended user attributes, in some cases needed for extended file options

barrier=1       1 = enabled. "Enforces proper on disk ordering of journal commits,
                makes volatile disk write chaches safe to use" (from man page mount)

data=ordered    "All data is forced directly out to the main file system prior to
                its metadata being committed to the journal" (from man page mount)
```

This is an abstract of how a command looks like when mounting a device (volume) - you can find those commands in the Command Log panel which is also described above:

```
> "/bin/mount" "-t" "ext4" "/dev/vgfaithdata/demo" "/media/demo"
```

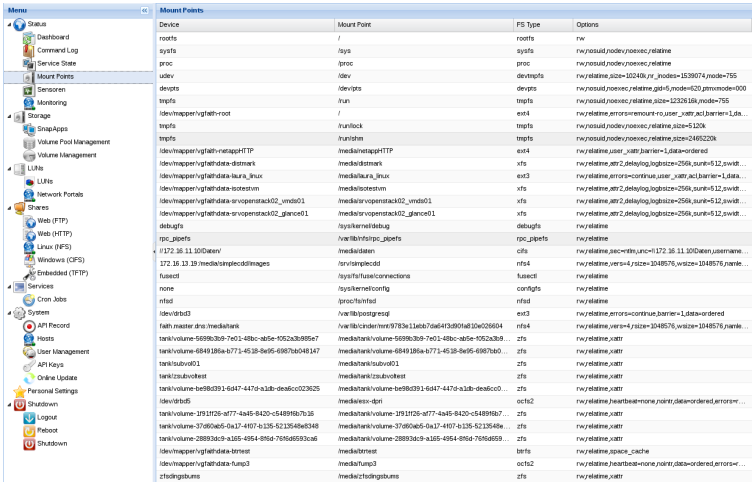


Figure 6.4: Mount Points panel

6.1.5 Sensors

openATTIC also takes care of the system's hardware - the Sensors panel offers a list of important hardware components and their state and current value (degrees / volts / rpm). If for example the CPU temperature is too high or a fan doesn't work anymore you will see it here, so you're able to take measures against it.

6.1.6 Monitoring

OpenATTIC is not only able to deal with storage issues. By integrating the monitoring software Nagios® it is also able to give you additional information about the storage, as well as the system's health. You don't have to add checks manually - for example by creating a volume a nagios service will be created automatically as well as checks for important openATTIC services like `openattic_rpcd` and `openattic_systemd`. The monitoring panel gives you an overview of all service checks and their current states as well as location of the service (resource host), date and time of last and next checks. It keeps you up to date so you can react for example if there is not much space left on a disk so you can resize it or a service is down so you can start it again.

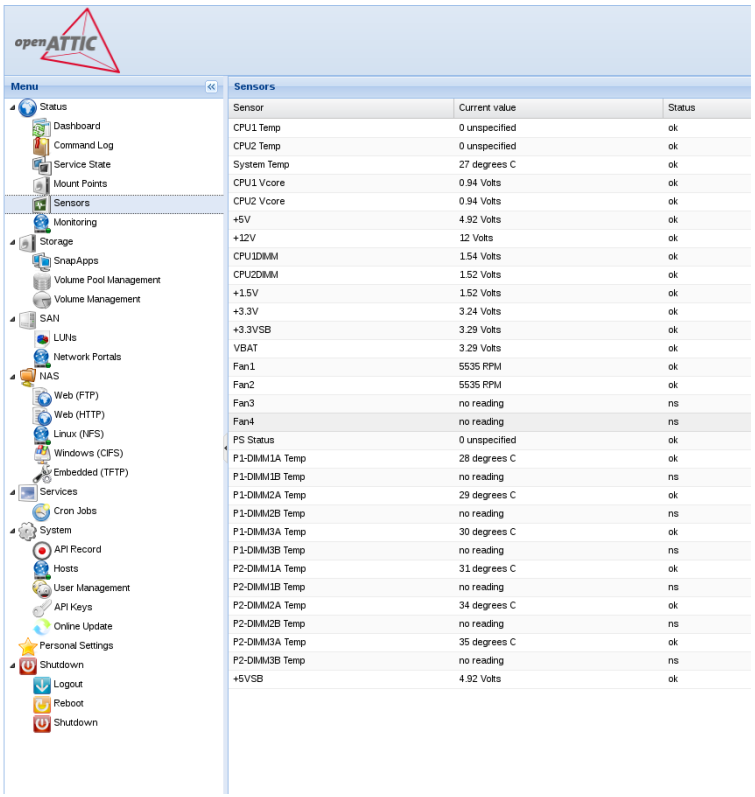


Figure 6.5: Sensors panel

By entering your e-mail address in the user management panel you will get notifications, for example when a service is in a critical state - here is an example:

```
***** Nagios *****

Notification Type: PROBLEM

Service: Disk Space
Host: localhost
Address: 127.0.0.1
State: CRITICAL

Date/Time: Thu Oct 31 09:54:00 CET 2013

Additional Info:

DISK CRITICAL - free space: /media/daten 579849 MB (10% inode=99%):
```

By selecting a service check you can see relevant graphs of 4 hours, 1 day, 1 week, 1 month and 1 year.

This is where you add graphs to your dashboard:

> Select the service you want to add to the dashboard and click the "Dashboard" button. You can rename the suggested graph name if you want to and click "ok" and your graph will appear on the Dashboard.

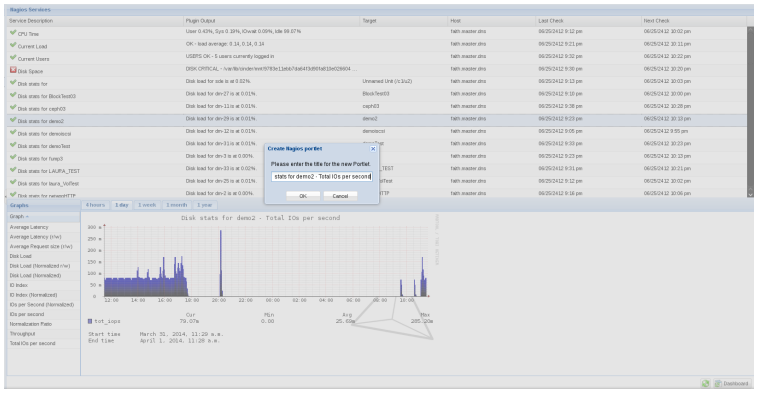


Figure 6.6: Add graph to dashboard

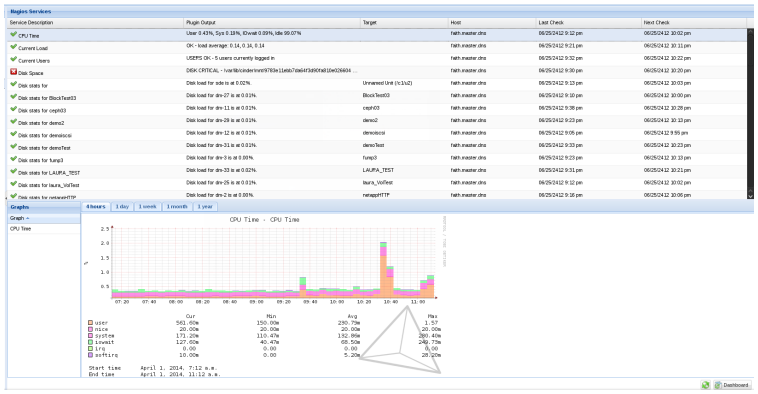


Figure 6.7: Monitoring panel overview

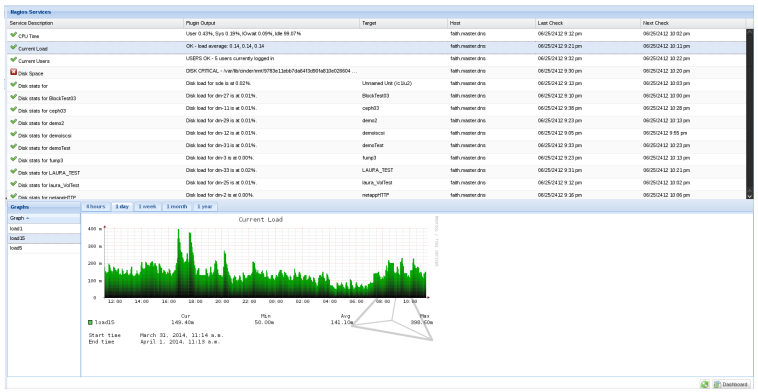


Figure 6.8: Monitoring panel overview

6.2 Storage

6.2.1 Volume Pool Management

The Volume Pool Management panel displays not only all volume groups but structure based information like a raid and disks within a volume group, if there are sub-devices beneath a volume group - you can see a "+" in front of the volume group. You can see detailed information about the volume groups as well:

- Type of the listed item (volume group, zpool, array, raid)
- The size of the volume groups, raids, arrays, pools and disks
- Free and used space of the volume groups (in GiB and percent)
- The state of the volume groups, pools, raids and disks

In the image below you can see that "vgfaithdata" consists of a raid 0 (named "MDRAID array md0"). The raid 0 again is splitted into two raid 5. Each raid 5 ("raid2r", "raid0r") consists of five disks (see "disk object"). You can also see the that the type of the disks is "SAS 15k" which means that those are SAS (Serial attached SCSI) hard drive disks with 15000 rpm (rotational speed). 15 k is a performance indicator - the higher the revolution per minute, the faster is the disk.

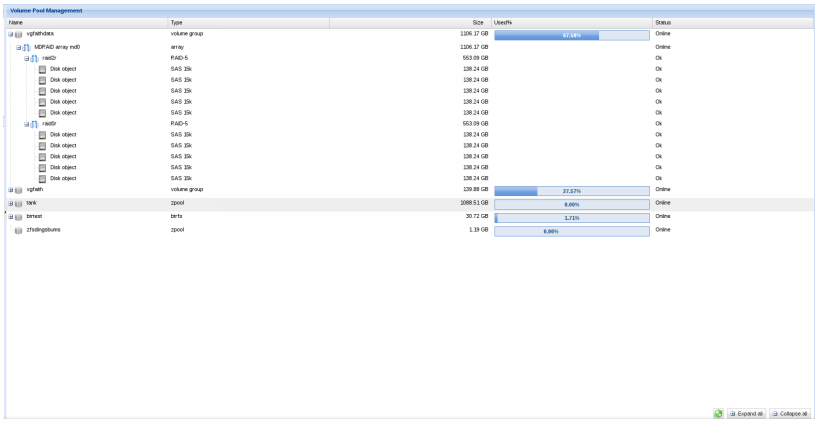


Figure 6.9: Volume Pool Management panel

How to add a volume group to openATTIC:

- `oaconfig add-vg` - adds an existing volume group to the openATTIC database
- `oaconfig add-disk` - formats a disk and adds it to a volume group.

If the volume group doesn't exist yet, it will be created and added to the database

6.2.2 Volume Management

As its name says the Volume Management Panel manages anything that has something to do with volumes. Every volume group and the volume which belongs to that volume group are listed.

There are also more details about each item in the list, like: * Type (either volume group or the file system, else you will see a "-" for block volumes) * Size * Used space in percent (depending on the set warning and critical level the status bar will turn orange / red when the level is achieved) * Status (online, ok, offline) * Warning level in percent

* Critical level in percent * Path (i.e., /dev/vgfaithdata/test) * The owner of the volume * Free space of the volume * Used space of the volume

The following actions are supported:

- Add Volume(s)
- Delete Volume(s)
- Edit Volume(s)
- Resize Volume(s)
- Snapshot Volume(s)
- Mount Volume(s)
- Unmount (s)
- Mirror Volume(s)
- Reload Volume list

Note: A more detailed explanation of the above mentioned actions can be found below (see: Volume Management Actions).

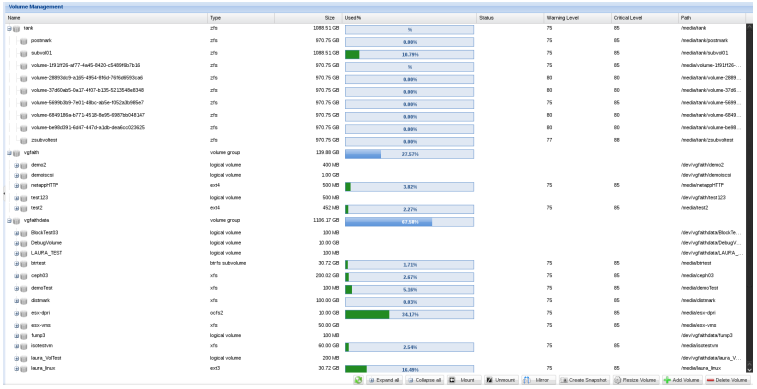


Figure 6.10: Volume Management panel

Depending on what you want to use a volume for, you have to decide how to configure it. You have to consider some points, if you care about performance or high-availabilty of your data, for example.

There are some use cases annotated which should help you to make the right choice. You can use openATTIC's storage for a *fileserver*, as storage for *virtualization*, providing *cloud storage* and *more*. If you've already decided what you want to do, you may need a litte help to implement it - here are some step by step *instructions*.

Supported file systems

ZFS (Z File System)

- Combined file system and logical volume manager
- support for high storage capacities
- efficient data compression
- integration of filesystem and volume management
- snapshots, copy-on-write clones

- continous integrity checking automatic repair
- uses a software data replication model (RAID-Z) and native NFSv4 ACLs
- max. file size: 16 EB
- max. supported volume size: 256 ZB

ext3 (third extended filesystem)

- journaled file system (improves reliability and eliminates the need to check the file system after unclean shutdown)
- commonly used by linux systems
- uses less CPU power than XFS
- max. file size: 2 TB
- max. partition size: 32 TB

ext4 (fourth extended file system)

- supported volume size up to 1 exbibyte (EiB
- supported file size up to 16 tebibytes (TiB)
- improve large file performance, reduce fragmentation

xfs

- high performance 64-bit journaling file system
- excels in parallel input/output operations due to its design
- enables extreme scalability of I/O threads, file system bandwith, and size of the file system itself
- We recommend using XFS for *virtualization*
- max. file size up to 8 EB
- max. partition size up to 8 EB

btrfs (B-tree file system)

- experimental
- online volume growth and shrinking
- Online balancing (btrfs moves objects between disks to balance the load)
- Subvolumes
- File cloning
- max. file size: 16 EB
- max. supported volume size: 16 EB

Volume Management Actions

Here you can find some basic steps (like creating, deleting, resizing, mirroring a volume) described in more detail.

This is where you can add a volume:

- Click the "Add Volume" button and enter the name of the new volume
- Choose the volume group in which the volume should be created

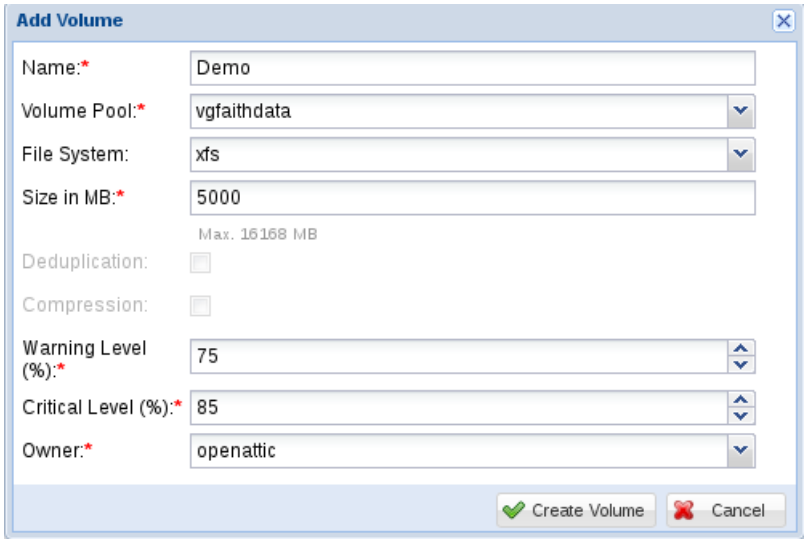
- Select a file system

Note: If you want to mirror the volume or share it using SAN protocols, do not create a file system on this volume.

See also:

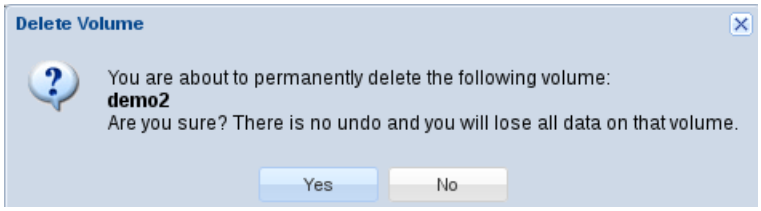
Mirror configuration (below)

- Enter the size
- Warning Level: default is at 75% but you can change it if you want to
- Critical Level: default is at 85% but you can change it as well
- Owner: choose the owner of the new volume
- Click "Create Volume"



Delete a volume:

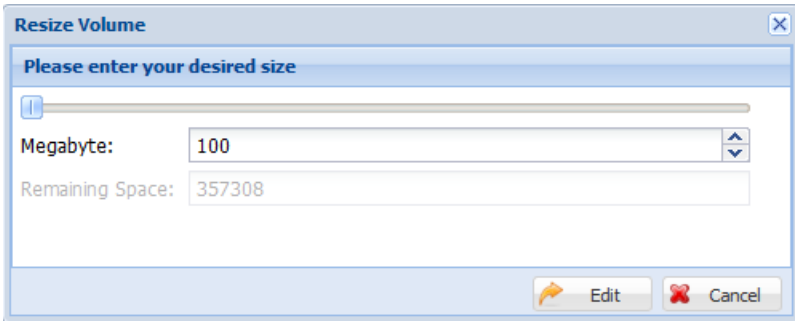
- Select the volume you want to delete in the volume list
- Click "Delete Volume" button
- Confirm


Delete Volume
You are about to permanently delete the following volume:
demo2
Are you sure? There is no undo and you will lose all data on that volume.
Yes
No

Resize a volume:

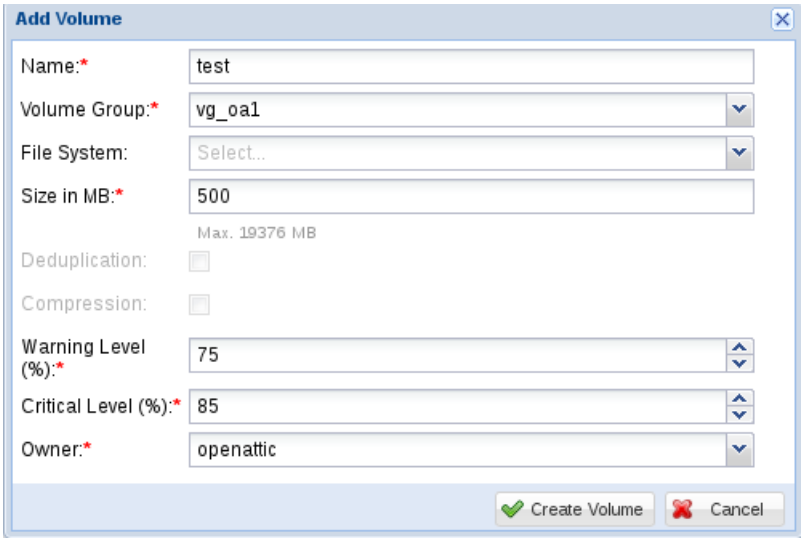
- Select the volume you want to resize in the volume list
- click "Resize Volume" button
- You can change the size by editing the value in the "Megabyte" field or using the scrollbar
- Click the "Edit" button

- Confirm your changes



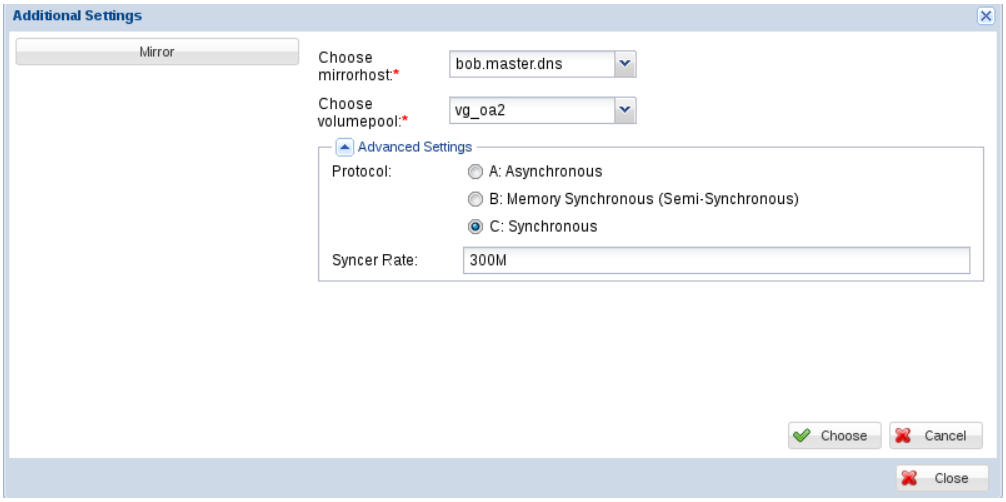
Basic steps for mirror configuration:

- **with a new volume:**
  - Click "Add Volume" button
  - Configure the volume properties - **leave file system field empty!**
  - Click "Create Volume" button

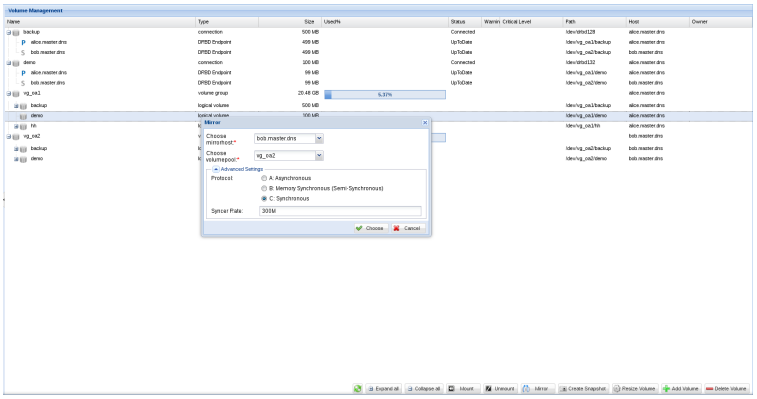


- Click the "Mirror" button
- Select the mirror host from the list of volumes
- Choose the volumepool on the mirror host into which the volume should be mirrored (the mirror-volume will be created automatically)
- Advanced options:
  - you can leave the standard here or choose between the mirror protocols:
  - **A: Asynchronous** This protocol is often used in long distance replication cases.
  - **B: Memory Synchronous (Semi-Synchronous)** Only synchronizes the network traffic (we recommend using protocol C instead).
  - **C: Synchronous** The most commonly used protocol. Data is fully synchronized on both nodes.
  - you can leave the standard here or configure the Syncer Rate

- Click “Choose” button and close the window

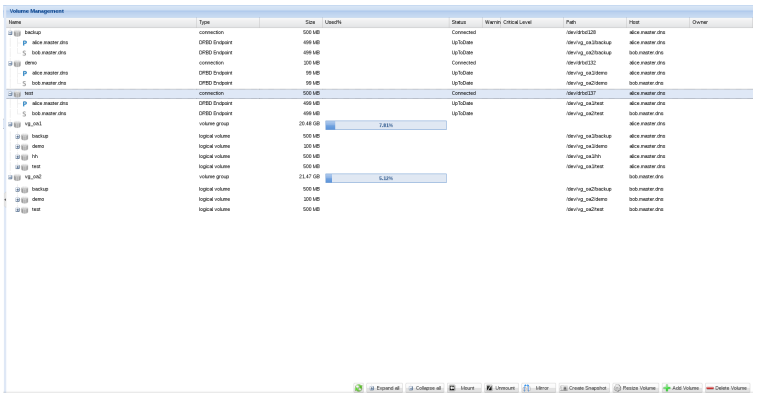


- **With an existing volume:**
  - Select volume which you want to mirror from the list of volumes
  - **Click “Mirror” button**
    - choose the mirror host
  - Choose the volumepool on the mirror host into which the volume should be mirrored (the mirror-volume will be created automatically)
  - Advanced options:
    - Choose between the mirror protocols:
    - A: Asynchronous
    - B: Memory Synchronous (Semi-Synchronous)
    - C: Synchronous
    - Configure Syncer Rate
  - Click “Choose” button and close the window



- **Mirror completion**

  Now you can see the volume in the list with the DRBD® endpoints - in this example the primary host (marked in the gui as "P") = alice, secondary host (marked in the gui as "S") is bob and their state ("UpToDate")



6.2.3 SnapApps

The openATTIC snapcore allows you to create snapshots of devices according to a schedule. It is part of the openATTIC community version and is also the base for openATTIC SnapApps which are provided in the openATTIC enterprise edition. The snapcore comes with a configuration wizard which helps you to configure scheduled snapshots step-by-step.

You can backup sensitive applications like databases or virtual machines as part of your nightly datacenter backup, but to do so, it sometimes requires some processes to be stopped and write access is not possible at that time.

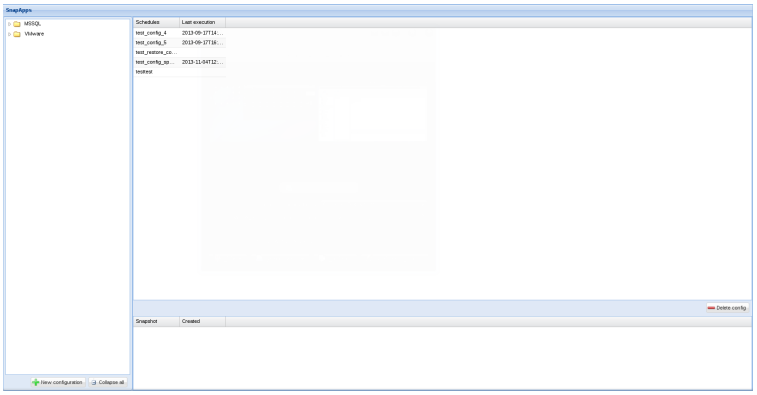
So a better solution is to use snapshots for applications. You can create snapshots from running systems and it only takes a few seconds or even milli seconds. When a snapshot of an application is created, the system will be brought into a consistent state by "freezing" it and not allowing write access, to make sure that the snapshot is not broken. After the snapshot of the application itself (software-layer) was taken, a snapshot of the volume (block-device layer), where the application is located, will be taken. Which means that the snapshot of the application itsself can be deleted again, because it's stored within the block device snapshot. By deleting the software layer snapshot you can ensure that there is no performance loss of the running system.

The following steps will show you how to deal with the configuration wizard and the different configuration options.

- **Available options:**
  - Add new configuration

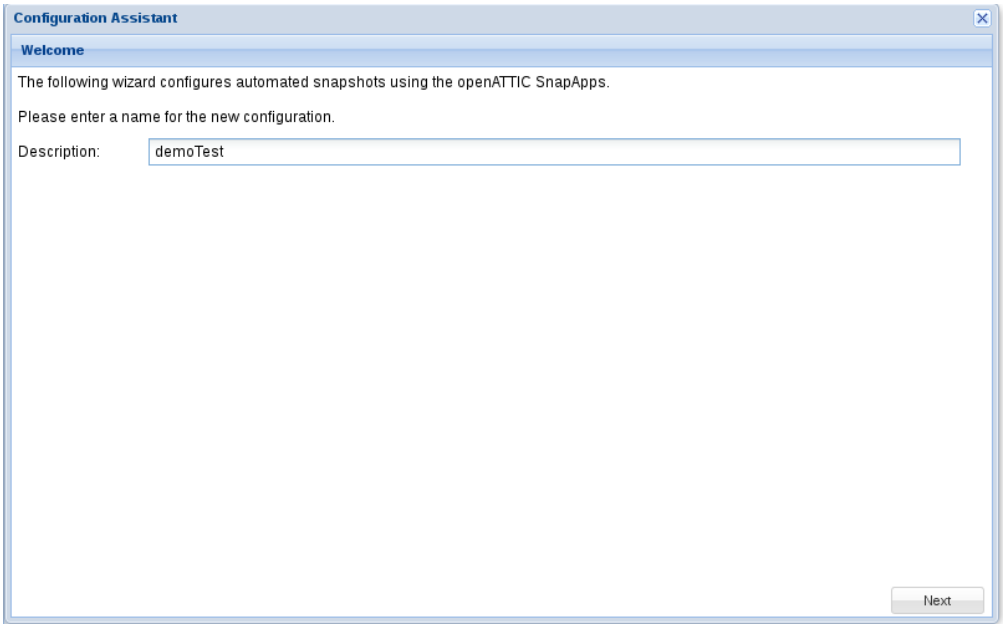
- Collapse all - hides all subitems of the installed SnapApp plugin folders (left side of the panel) i.e., MSSQL, VMware
- Delete config

The available SnapApp plugins are displayed on the left, in the right panel you can see existing configurations (schedules) and their last execution (date, time). By selecting a configuration in the list, you can see the list of created snapshots (name, create date) in the panel below.



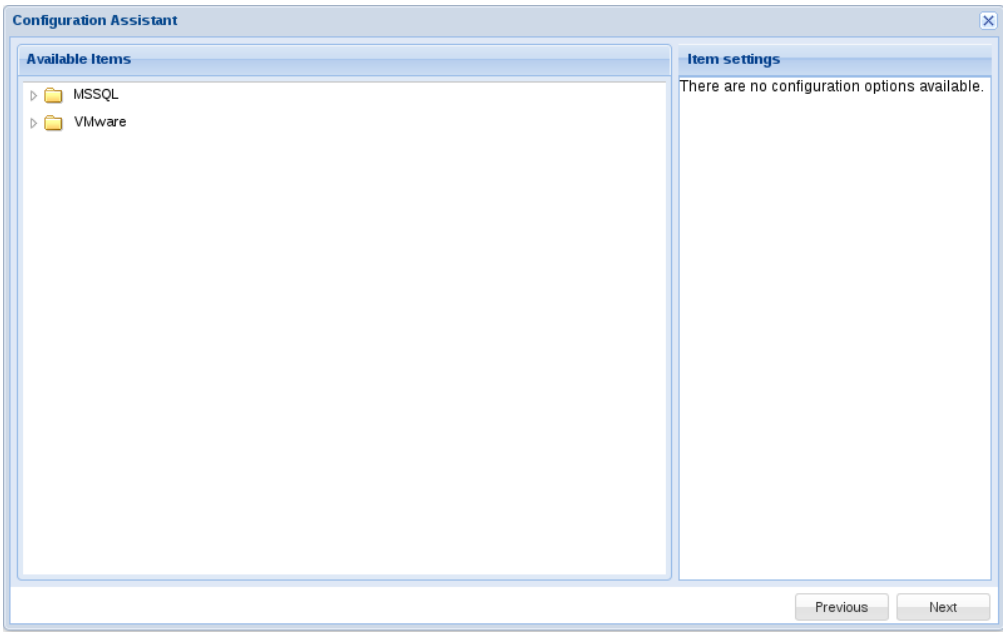
Wizard configuration - Step 1: Configuration Name

Start the wizard configuration by clicking the "Add configuration" button. Enter a name for your new config.



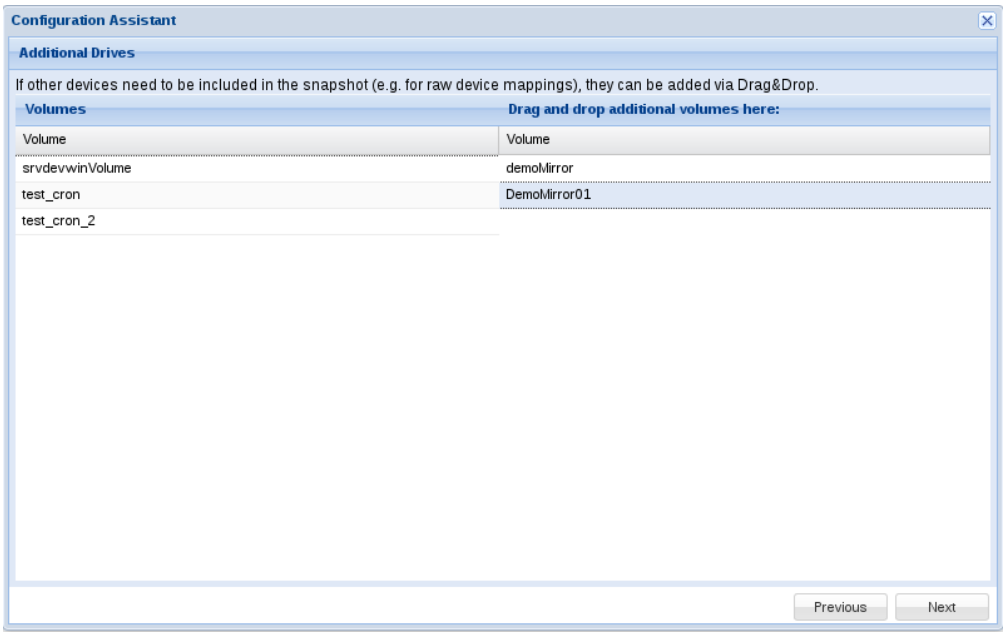
Wizard configuration - Step 2: Available Items

If you have one or more SnapApp-Plugin(s) installed, items available for snapshotting will be displayed here. You can select the items you want to snapshot and click "Save". If there aren't any Plugins installed you will see the message "There are no configuration options available" in the right panel of the window. So you can just skip this part.



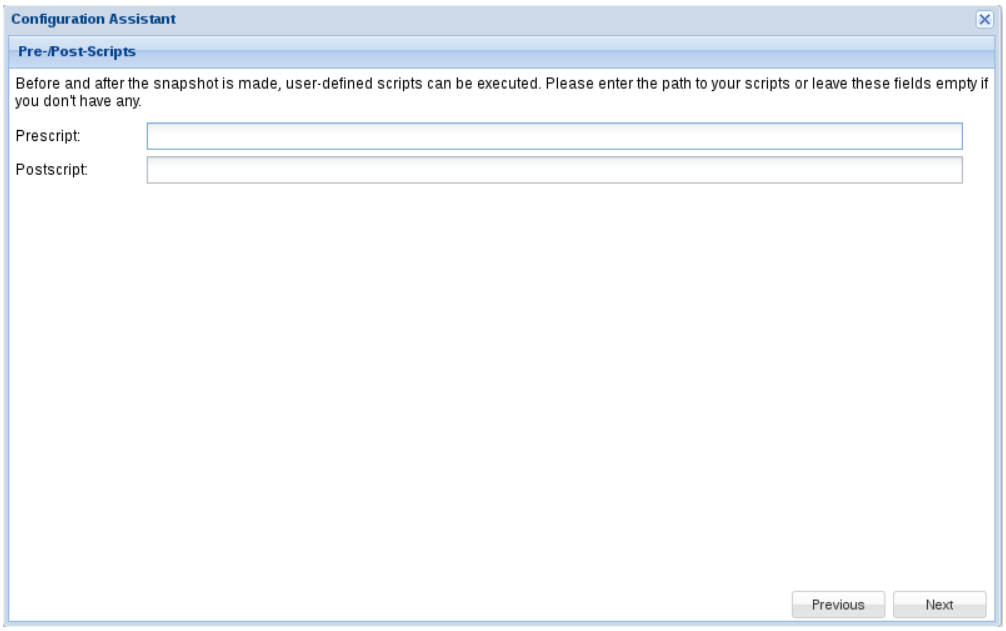
Wizard configuration - Step 3: Additional Drives

Choose additional drives to be snapshotted and drag them into the right half of the window.



Wizard configuration - Step 4: Pre-/Post-Script Conditions

If you want to execute any scripts before or after snapshotting, enter them here.


Configuration Assistant
Pre-/Post-Scripts
Before and after the snapshot is made, user-defined scripts can be executed. Please enter the path to your scripts or leave these fields empty if you don't have any.
Prescript:
Postscript:
Previous
Next

Wizard configuration - Step 5: Expiry

You can choose between snapshots without or with expiry date. Snapshots can be automatically deleted, just select the retention period in

- Seconds
- Minutes
- Hours
- Days
- Weeks

Wizard configuration - Step 6: Execution

Now you can configure the schedule options.

- Execute immediately (only once)
- Execute later at a specific date/time (only once)
- **Or create Schedule by**
  - Selecting the start date / start time
  - Choosing “end date / end time” or “No end date”
- Setting the schedule to “active”

Decide if your scheduled snapshot should only run once (immediately or at a given day/time) or if you want them to run for a specific period

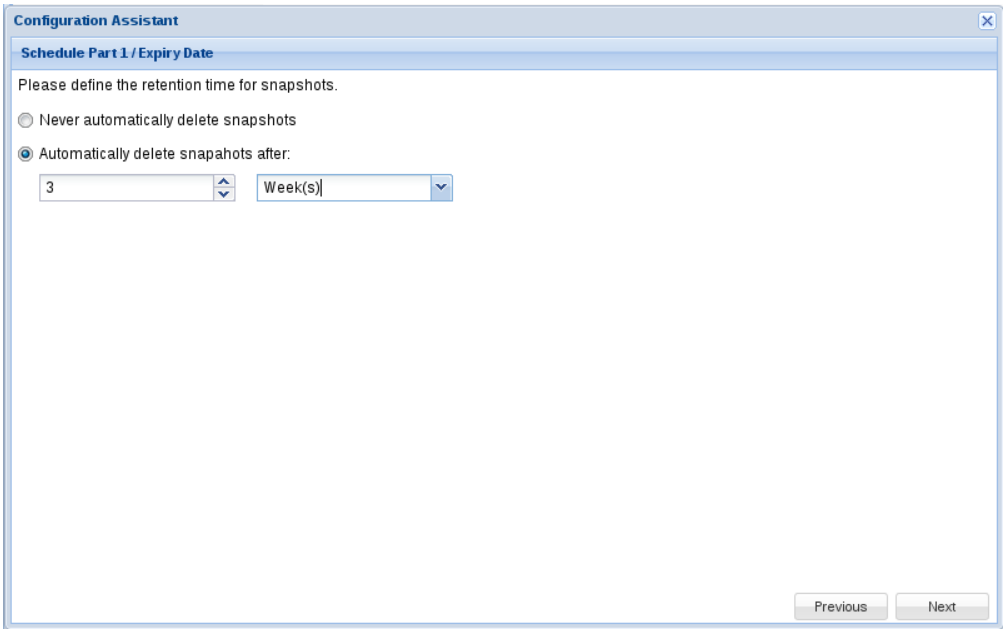


Figure 6.11: with expiry date

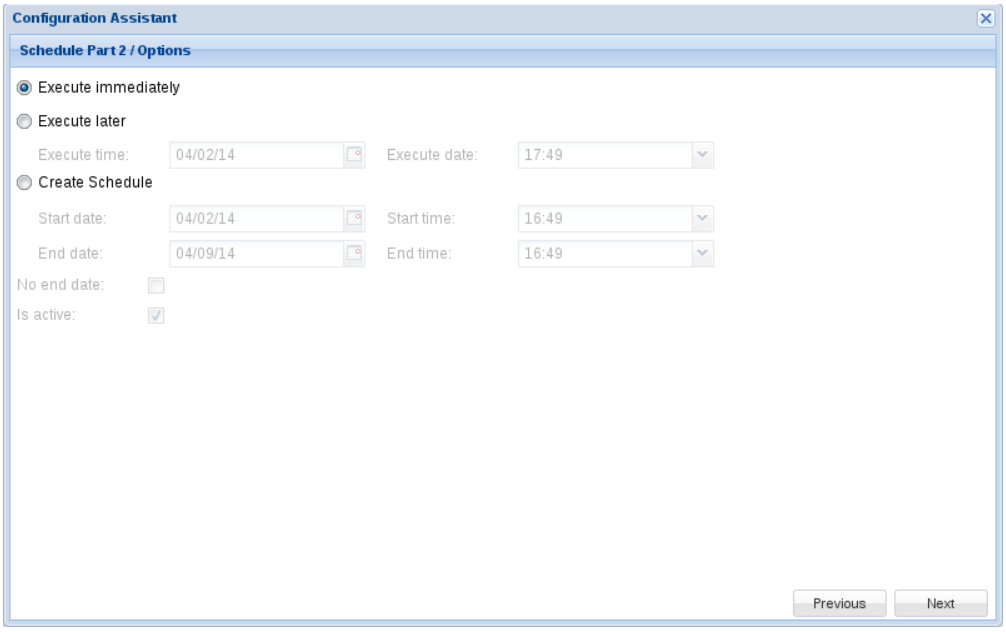


Figure 6.12: Will be executed immdediately

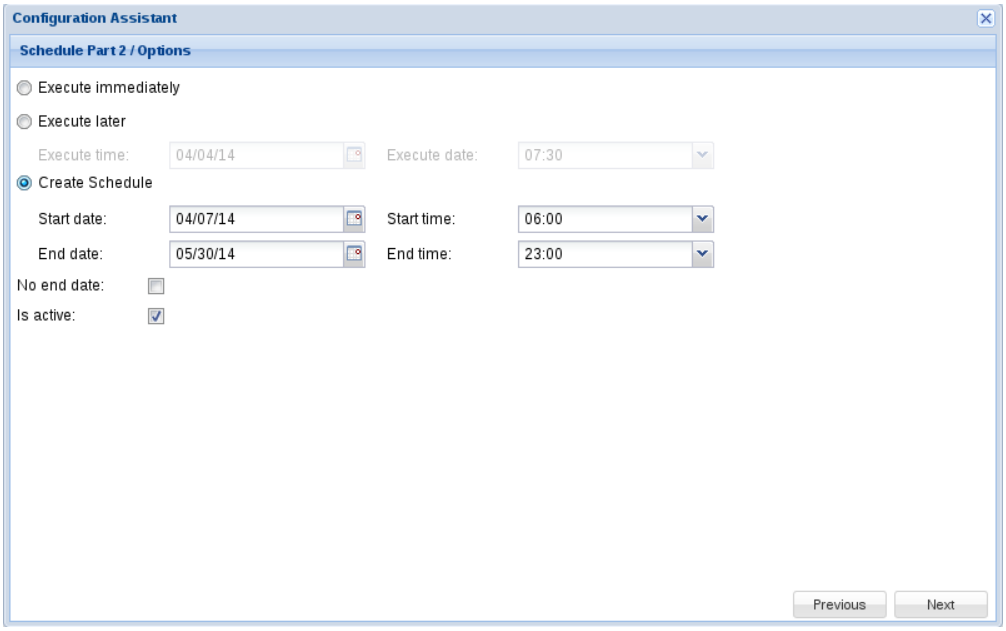


Figure 6.13: scheduled snapshots

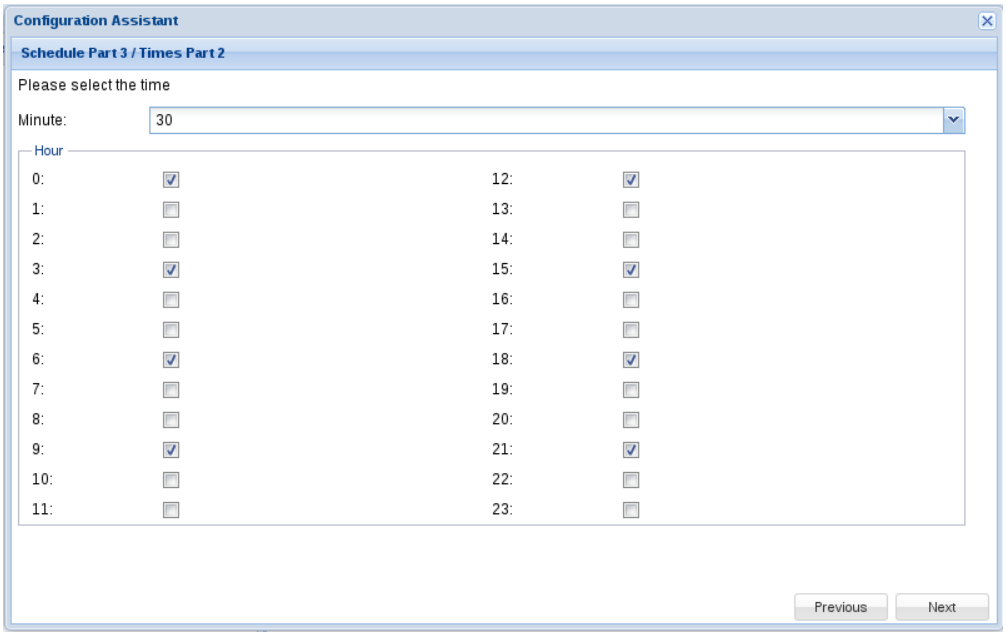


Figure 6.14: Scheduled snapshots: select the time interval in which snapshots should be taken

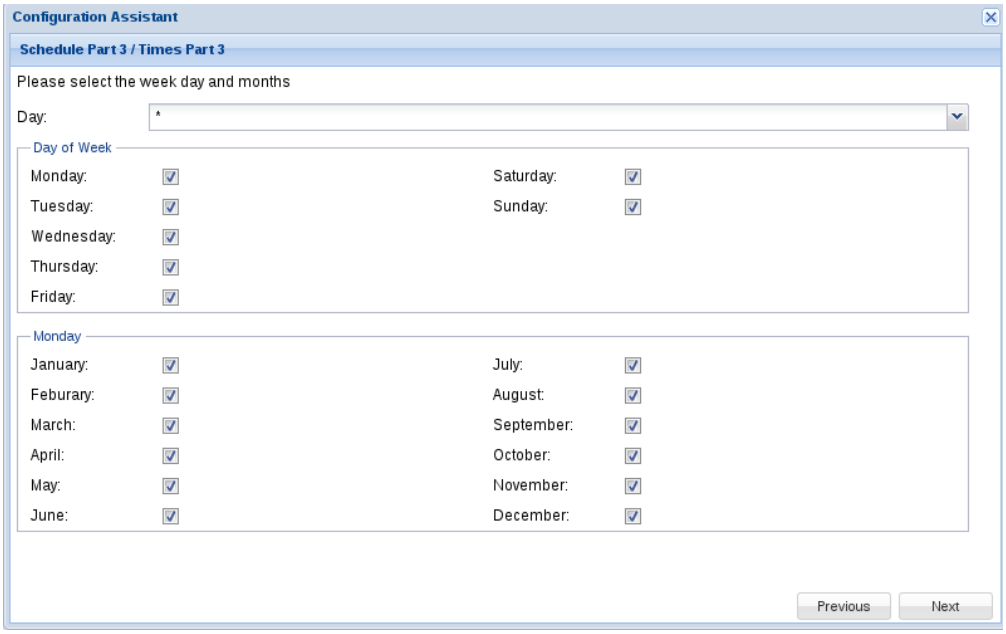


Figure 6.15: Scheduled snapshots: select the interval (days, weeks, months) in which snapshots should be taken

6.3 LUNs (SAN)

While *file shares* are adequate for many applications, there are others which require a more direct way of accessing storage. For instance, systems may rely on file systems that are not supported on the storage system directly (for instance, NTFS), or the application may need to employ block-device oriented technologies like MD-RAID or LVM.

Such applications are not well suited using the file-based approach that NAS protocols have to offer, and require a different strategy which is provided using SAN protocols.

6.3.1 Protocols

Basically, all SAN protocols are an extension of the SCSI protocol traditionally used for connecting controllers and disks, transporting SCSI commands and data over a copper or fiber-optical link to remote machines. This way, the same paradigms for accessing local disks also apply to accessing volumes on a remote storage system.

When dealing with SCSI setups, there is a bit of terminology that is useful to know:

- Controllers that wish to make use of a storage device are called *Initiators*. They are responsible for initiating SCSI sessions and taking the decision of when to send which commands.
- Storage devices are called *Targets*, because they are targeted by the Initiator.
- Initiators can be implemented both in software or in hardware. Hardware initiators are called Host Bus Adapters (*HBAs*), because they provide an interface between the Host's PCI bus and the SCSI target.

  Whether or not an HBA is available and/or necessary depends on the protocol used and the application: For FibreChannel, an HBA is required; for FCoE and iSCSI it is optional.
- Targets provide one or more Logical Unit Numbers (*LUNs*), which are used by the initiator to identify the storage volume targeted by the operation. This way, targets can provide multiple volumes to different Initiators.

The most significant difference between SAN protocols is their classification in the OSI model, meaning that different SAN protocols operate over different kinds of networks:

- *FibreChannel* is a data link (layer 2) protocol. It does not use any underlying network structure at all and requires its own infrastructure of switches (called *Fabric*).
- *FibreChannel over Ethernet* is a network (layer 3) protocol and operates on standard Ethernet networks.
- *iSCSI* is an application (layer 7) protocol and operates on standard IP networks such as the Internet.

Since all these protocols share the same paradigms, they can all be configured in one single interface in openATTIC.

6.3.2 Configuring openATTIC as a Target

In order to make use of SAN protocols, openATTIC needs to be configured as a Target first. How this is done depends on the protocol used.

FibreChannel

Note: In order to use FibreChannel you need at least the kernel version 3.5. When using Debian Wheezy (which currently uses the kernel version 3.2) you can install the kernel from the wheezy backports repository.

---

1. Make sure your openATTIC system is equipped with at least one QLogic FibreChannel HBA.
2. Install the `firmware-qlogic` package.
3. Configure the Linux FC driver to set your HBAs to Target mode.
4. Reboot the system for the changes to take effect.
5. Run `oaconfig install` in order to detect available FC targets.

Use the following commands to do this:

```
apt-get install firmware-qlogic
echo 'options qla2xxx qlini_mode="disabled"' > /etc/modprobe.d/qla2xxx.conf
update-initramfs -k all -u
reboot
oaconfig install
```

Please check supported adapters here.

iSCSI

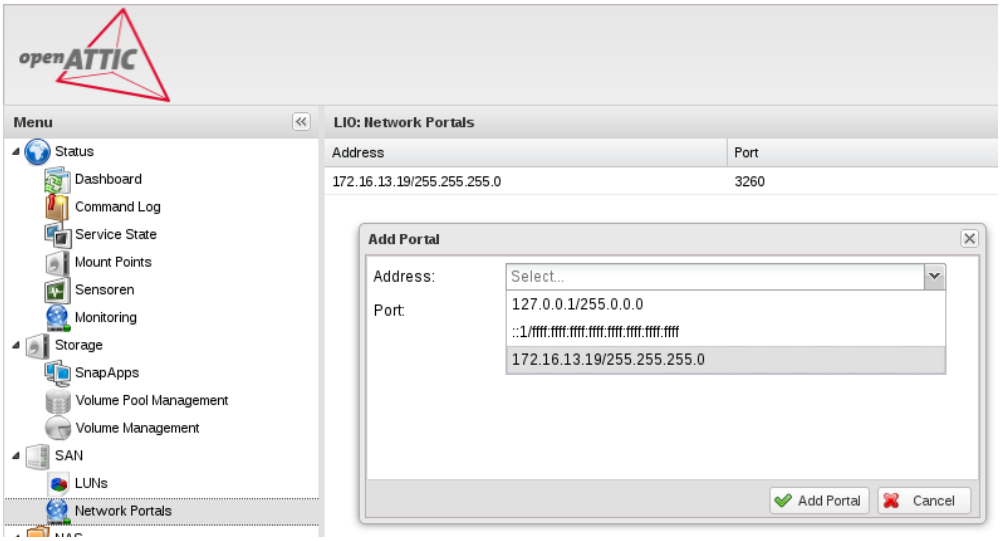
iSCSI requires the IP address to be configured, under which the target should be reachable to the outside world. This can be done in the Network Portals panel.

Note: The `oaconfig install` command creates a default network portal when scanning for the IP addresses of your openATTIC system.

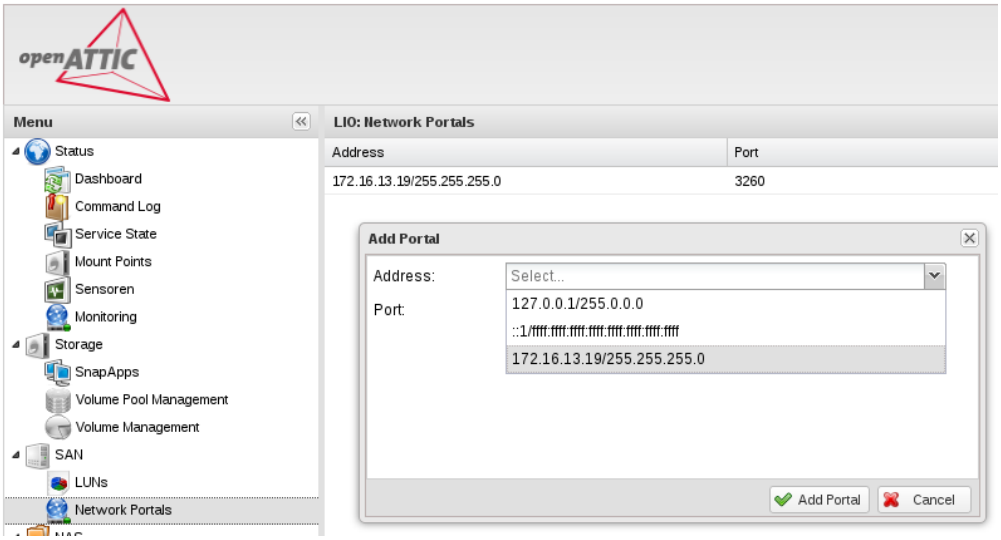
In order to add a portal, click the "Add Portal" button, select the IP address and click on "Add Portal". Only those IP addresses for which portals have been configured will respond to iSCSI connections.

6.3.3 Configuring Initiators

If a volume is to be available via SAN protocols, a LUN needs to be configured for it. However, openATTIC hides the complexity of configuring the relevant bits of information for different storage protocols, so the same volume can easily be configured for both FC and iSCSI protocols if desired. In order to do this, openATTIC needs to know which



hosts are capable of being initiators, and which protocols they are able to use. This information is configured in the Hosts panel under "Attributes".



To configure a host as an initiator:

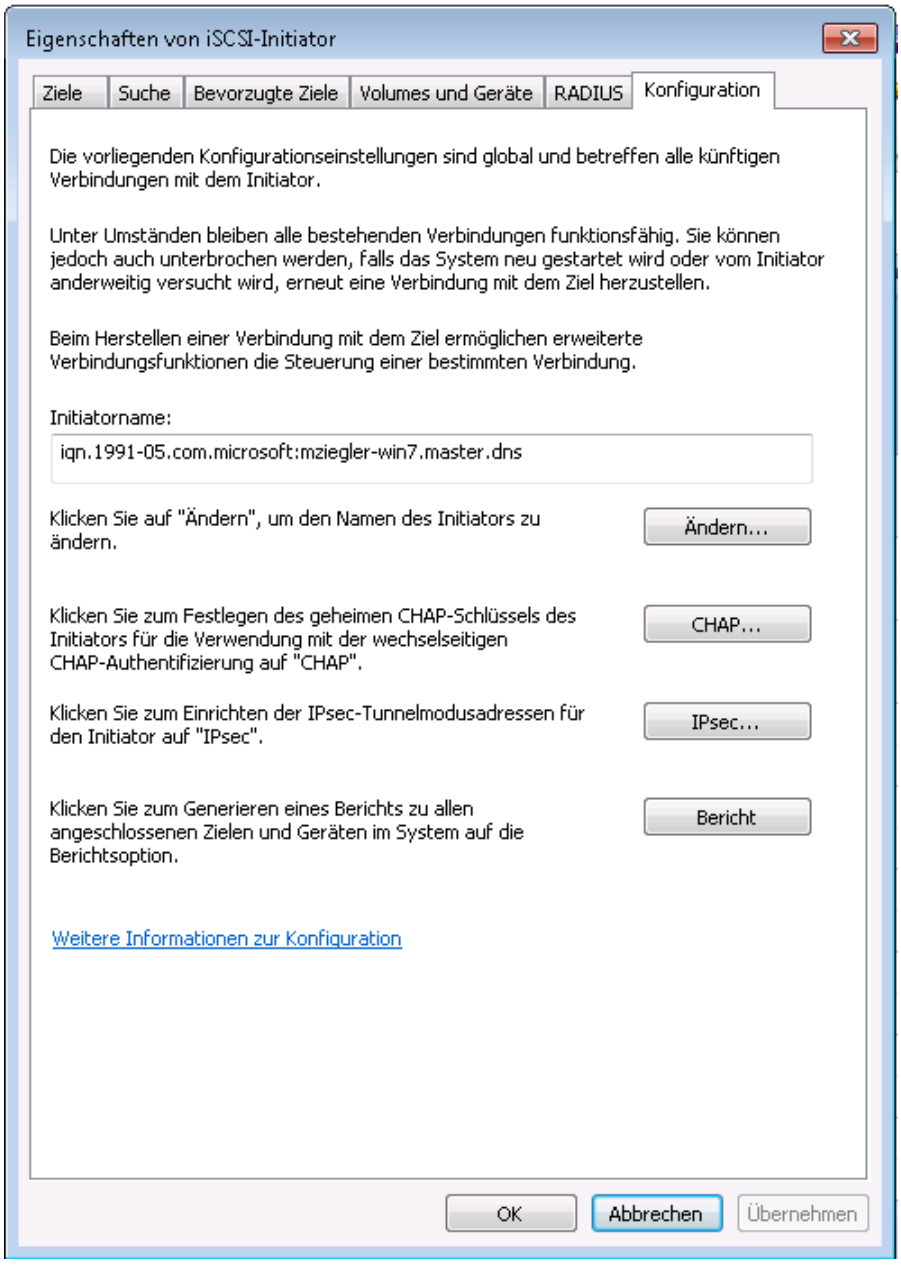
1. Add the host if it does not exist already by clicking the "Add Host" button and entering its fully-qualified domain name.
2. Select the host to see its attributes.
3. Select the "Initiators" entry on the right hand side.
4. Click "Add attribute".
5. Choose the storage protocol supported by your initiator.
6. Depending on the protocol, enter the FibreChannel WWN or iSCSI IQN of your host.
7. Click "submit".
8. The "Initiators" subsection now shows the initiator entry.

Finding the IQN

For iSCSI connections, you will need to know the IQN of the Initiator.

Windows

On Windows systems, the control panel offers an item called “iSCSI-Initiator”, where you will find the IQN on the “Configuration” tab:


Eigenschaften von iSCSI-Initiator
Ziele
Suche
Bevorzugte Ziele
Volumes und Geräte
RADIUS
Konfiguration
Die vorliegenden Konfigurationseinstellungen sind global und betreffen alle künftigen Verbindungen mit dem Initiator.
Unter Umständen bleiben alle bestehenden Verbindungen funktionsfähig. Sie können jedoch auch unterbrochen werden, falls das System neu gestartet wird oder vom Initiator anderweitig versucht wird, erneut eine Verbindung mit dem Ziel herzustellen.
Beim Herstellen einer Verbindung mit dem Ziel ermöglichen erweiterte Verbindungsfunktionen die Steuerung einer bestimmten Verbindung.
Initiatorname:
iqn.1991-05.com.microsoft:mziegler-win7.master.dns
Klicken Sie auf "Ändern", um den Namen des Initiators zu ändern.
Ändern...
Klicken Sie zum Festlegen des geheimen CHAP-Schlüssels des Initiators für die Verwendung mit der wechselseitigen CHAP-Authentifizierung auf "CHAP".
CHAP...
Klicken Sie zum Einrichten der IPsec-Tunnelmodusadressen für den Initiator auf "IPsec".
IPsec...
Klicken Sie zum Generieren eines Berichts zu allen angeschlossenen Zielen und Geräten im System auf die Berichtsoption.
Bericht
Weitere Informationen zur Konfiguration
OK
Abbrechen
Übernehmen

Linux

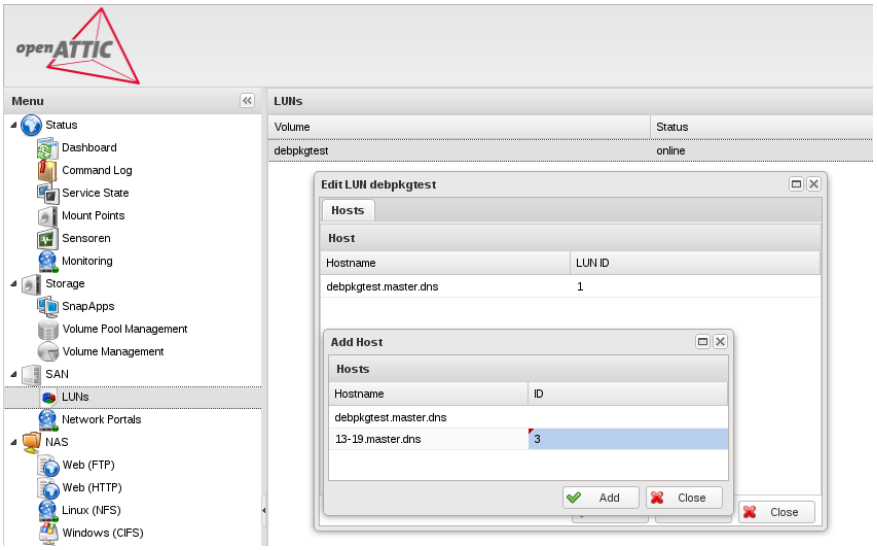
On Linux, you can find the IQN in the file `/etc/iscsi/initiatorname.iscsi` after installing and starting the `open-iscsi` service.

6.3.4 Configuring Volumes (LUNs)

openATTIC completely hides the complexity of configuring different SAN protocols for different hosts, depending on which protocols the target hosts support. For this reason, the GUI only asks you to select the hosts for which you would like a volume to be accessible, and then decides itself which protocols are appropriate.

Note: If an initiator supports multiple protocols, openATTIC will configure targets for all of them. The choice which protocol to use is up to the Initiator.

In order to configure a LUN, open the LUNs panel, select the volume you want to make available, and click the "Edit LUN" button. You will see a list of hosts that already have access to the volume.



To add a host, click the "Host list" button at the bottom of the panel. Enter a LUN ID in the column on the right, next to the host you wish to add, and click "Add". The host will be added to the volume's access list and openATTIC will automatically configure the necessary targets in the background.

Note: If unsure about the LUN ID, choose 1.

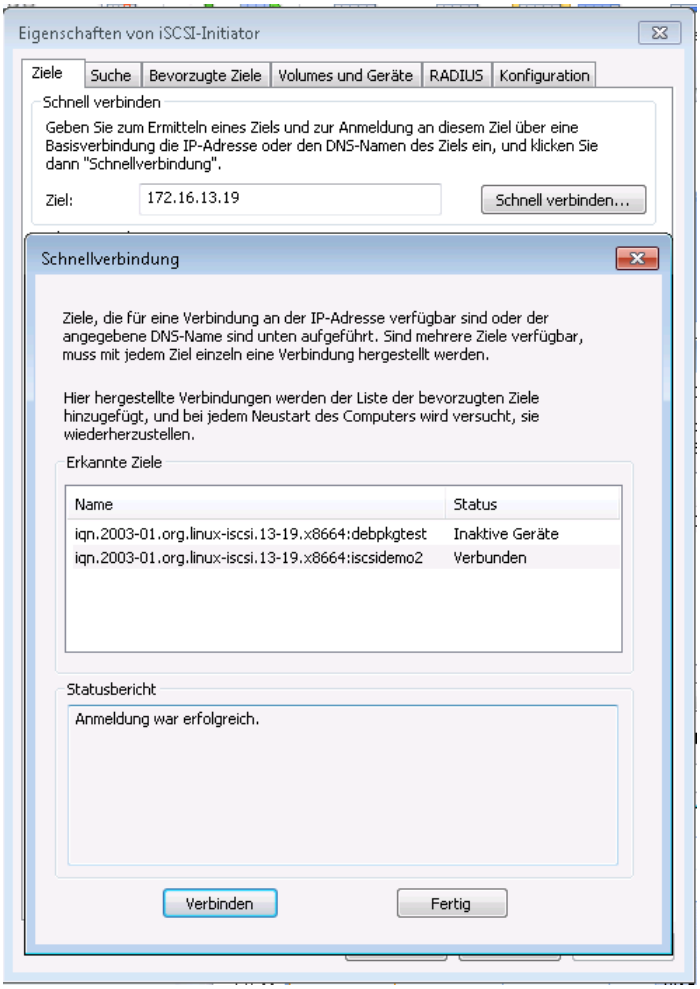
6.3.5 Using iSCSI volumes

To make use of the volumes you configured, you need to configure the initiator on the clients. How to do that, depends on the operating system.

Windows

On Windows systems, the control panel offers an item called "iSCSI-Initiator". The first tab allows you to discover and configure iSCSI targets. Enter the IP address of the openATTIC host into the box labeled "Target" and click the

"Quick connect..." button.


Eigenschaften von iSCSI-Initiator
Ziele
Suche
Bevorzugte Ziele
Volumes und Geräte
RADIUS
Konfiguration
Schnell verbinden
Geben Sie zum Ermitteln eines Ziels und zur Anmeldung an diesem Ziel über eine Basisverbindung die IP-Adresse oder den DNS-Namen des Ziels ein, und klicken Sie dann "Schnellverbindung".
Ziel:
172.16.13.19
Schnell verbinden...
Schnellverbindung
Ziele, die für eine Verbindung an der IP-Adresse verfügbar sind oder der angegebene DNS-Name sind unten aufgeführt. Sind mehrere Ziele verfügbar, muss mit jedem Ziel einzeln eine Verbindung hergestellt werden.
Hier hergestellte Verbindungen werden der Liste der bevorzugten Ziele hinzugefügt, und bei jedem Neustart des Computers wird versucht, sie wiederherzustellen.
Erkannte Ziele
Name
Status
iqn.2003-01.org.linux-iscsi.13-19.x8664:debpkgtest
Inaktive Geräte
iqn.2003-01.org.linux-iscsi.13-19.x8664:iscsidemo2
Verbunden
Statusbericht
Anmeldung war erfolgreich.
Verbinden
Fertig

The initiator will show you a list of discovered targets, most of which will be labeled inactive. Select the target you would like to connect to and click "Connect". The initiator will connect to the target and make its disks available to the system.

However, in order to actually use it, you will have to create a file system on the disk. To do this, open the system's "Disk Management" dialog, which you will find in the computer management utility. It will show you a new, unused disk. Right-click the disk and create a volume on it in order to make it available under some drive letter in the My Computer utility.

Linux

On Linux, `open-iscsi` can be used to connect to iSCSI volumes. To do so, perform an iSCSI discovery and a login as follows, substituting `172.16.13.19` with the address of your openATTIC host:

```
iscsiadm -m discovery -t st -p 172.16.13.19
iscsiadm -m node -l
```

The `lsscsi` utility will now display a disk by the vendor "LIO-ORG", which is your iSCSI connection:

```
# lsscsi
[1:0:0:0]    cd/dvd  QEMU     QEMU DVD-ROM     1.7.  /dev/sr0
[2:0:0:1]    disk    LIO-ORG  IBLOCK           4.0   /dev/sda
```

You can now format this volume with a file system, partition it or put it into an LVM volume group, just like any local disk.

6.4 Shares

File shares provide a means to access files on a remote system in the same way that you access files on the local disks. Depending on the operating system and protocols used, users won't even notice the difference because the file share seamlessly integrates into the system.

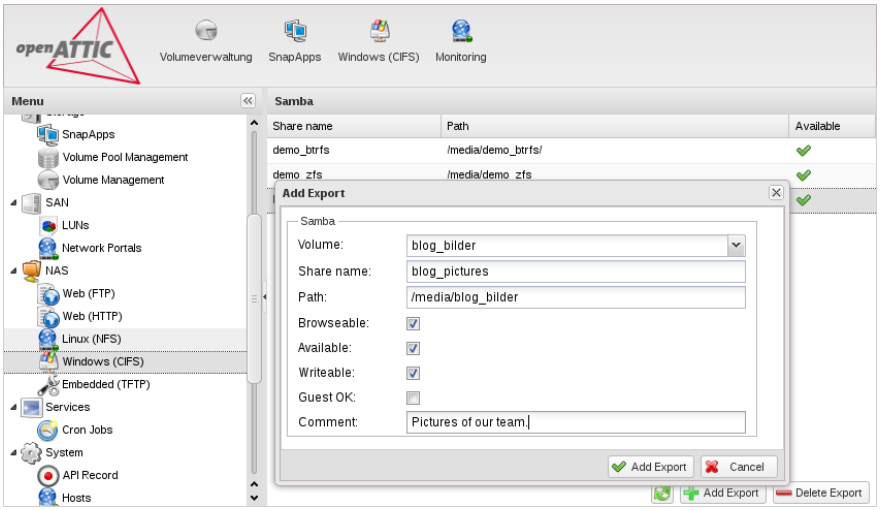
In this scenario, the actual file system is running on the storage server. This means that the storage can provide advanced functionality like caching, snapshots, deduplication and backups. Of course, each protocol is better suitable for some applications than for others – the module descriptions outline what each protocol is best used for, and link to use case implementation guides which describe the protocols in the context of their application.

6.4.1 CIFS

See also:

This step is part of *Implementing a file server*.

CIFS, formerly called Server Message Block (SMB), is a protocol established by Microsoft that is optimized for building central file storage servers used throughout the organization. This protocol has its strengths in user authentication, single-sign-on and authorization management.



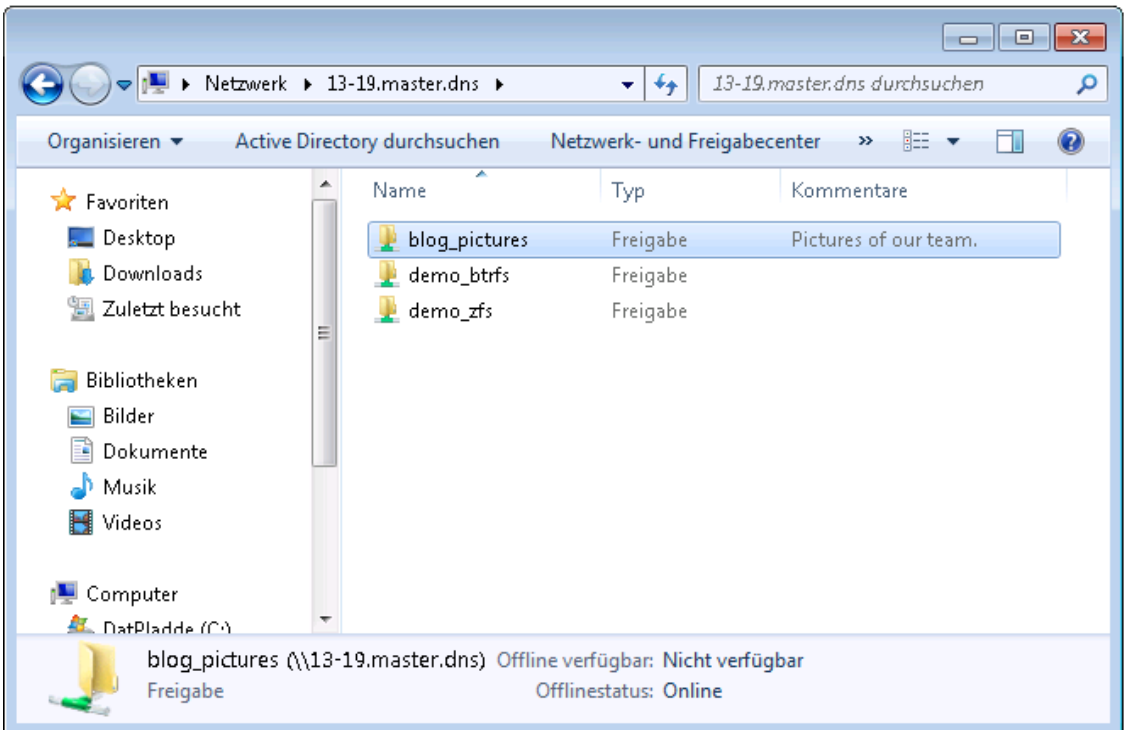
The CIFS panel allows you to share volumes using CIFS. To create a share, click the `Add Export` button, and provide the following information:

1. Choose the volume you would like to share.
2. Provide a name which will be visible to clients accessing the share. openATTIC suggests the shared volume's name, but you are free to change it.

3. If you would like to share only a certain subdirectory of the volume, you can enter its path in the `path` field. By default, this field is populated with the root directory of the volume.
4. To hide the volume from the list that appears in the Windows network browser, uncheck the `browseable` option.
5. To disable the share for a while, you can uncheck the `available` option.
6. To prevent all write access to the volume, uncheck the `writeable` checkbox.
7. To disable authentication, thereby allowing access to *any* user, you can check the `Guest OK` option.
8. The `comment` field will be shown in Windows Explorer's detail view and can contain a note to the user.

Note: The comment is *only* visible in the detailed view of Windows Explorer, which is not enabled by default. Do not rely on the information put in this field to be actually seen by your users.

Finally, click the `Add Export` button to create the new export which is instantly visible when browsing the openATTIC host in Windows Explorer.



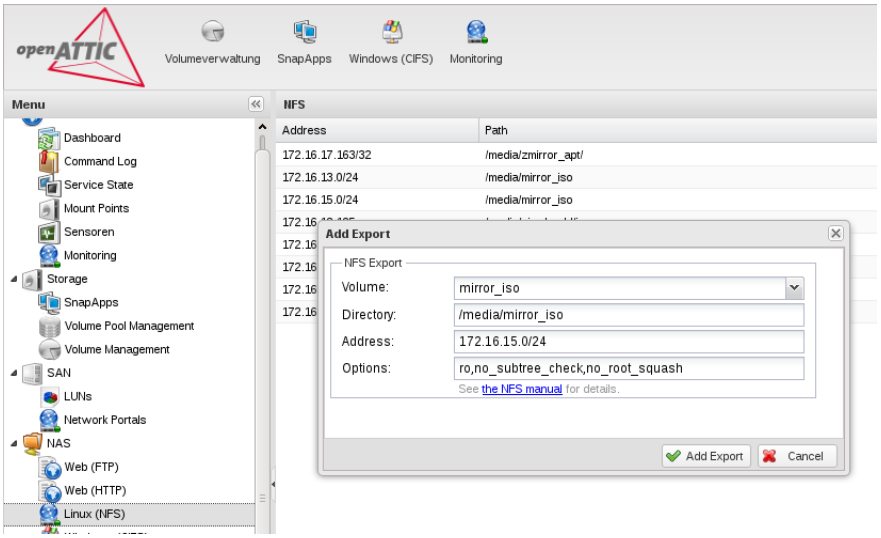
6.4.2 NFS

See also:

This step is part of *Implementing a virtualization storage*.

Network File System, in short NFS, is a protocol that allows Unix-like systems to share files between one another and has been originally developed by Sun Microsystems in 1984. This protocol seamlessly integrates remote file systems, providing access to the files stored within as if they resided on the local system. Its strength is the compliance to POSIX standards which makes it ideal for providing basic infrastructures.

The NFS panel allows you to share volumes using NFS. To create a share, click the "Add Export" button, and provide the following information:



1. Choose the volume you would like to share.
2. If you would like to share only a certain subdirectory of the volume, you can enter its path in the `path` field. By default, this field is populated with the root directory of the volume.
3. Enter the IP address(es) of hosts which should be allowed to mount the exported volume. This field can contain
   - a hostname
   - an IP address
   - a subnet specified as `address/CIDR" (e.g., ``172.16.15.0/24`)
4. Mount options. The defaults should be fine for most situations, but if your clients require special options to be set, you can do so here.

Finally, click the `Add Export` button to create the new export.

6.4.3 HTTP

See also:

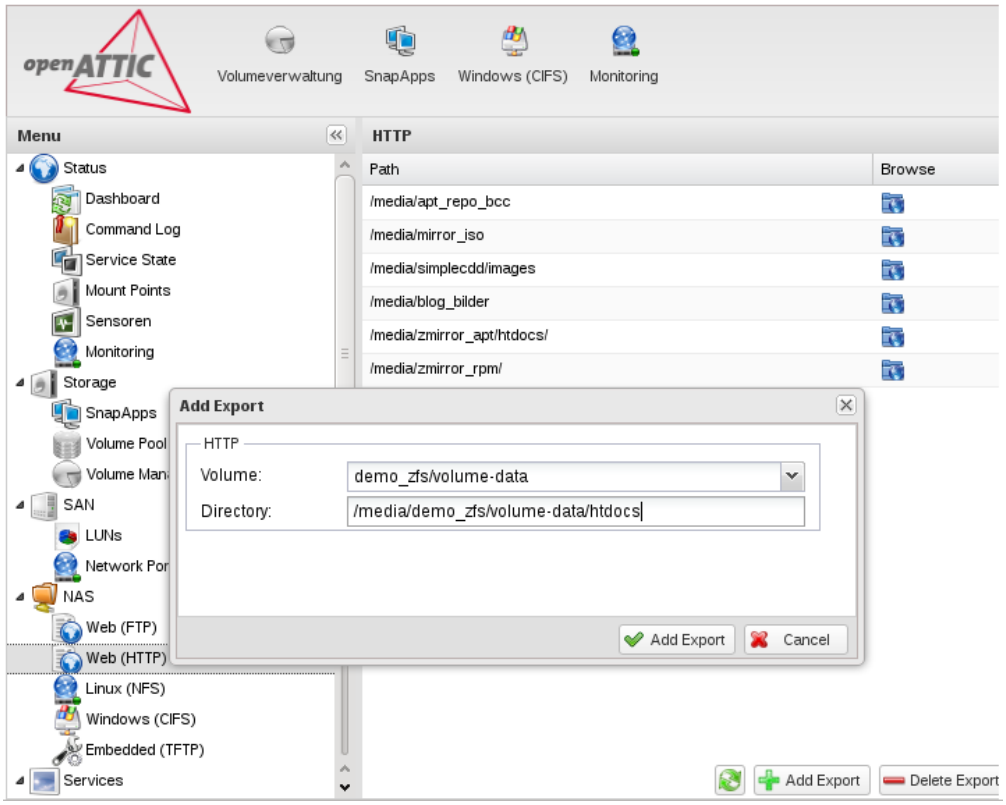
This step is part of *Mirror servers*.

Being the foundation of the World Wide Web, the Hypertext Transfer Protocol has been the cornerstone of information transfer between all kinds of different systems. It is supported by every operating system from standard computers, embedded or mobile devices to large-scale industrial installations, and handles the transport of huge – even endless – files just as well as tiny bits of text.

The HTTP panel allows you to export volumes using HTTP. To create an export, click the "Add Export" button, and provide the following information:

1. Choose the volume you would like to share.
2. If you would like to share only a certain subdirectory of the volume, you can enter its path in the `path` field. By default, this field is populated with the root directory of the volume.

Note: Exporting only a subdirectory makes sense for HTTP, because this allows you to customize the appearance and layout of the data inside the HTTP share independently from the way the data is actually stored. For example, you can export a Debian mirror while hiding the scripts that you use to manage the mirror by exporting a subdirectory and linking the mirror directory there.



By convention, such subdirectories are named `htdocs`.

Finally, click the `Add Export` button to create the new export. Next to each export, the "Browse" column provides a link to the volume so you can easily view it in your web browser.

6.4.4 FTP

Optimized for transfer of large files, the File Transfer Protocol allows a bit more fine-grained control over the handling of file transfers. Clients are more focused on file transfer than web browsers are, allowing multiple transfers in parallel and transferring files in a background connection instead of blocking the control connection for the period of the transfer.

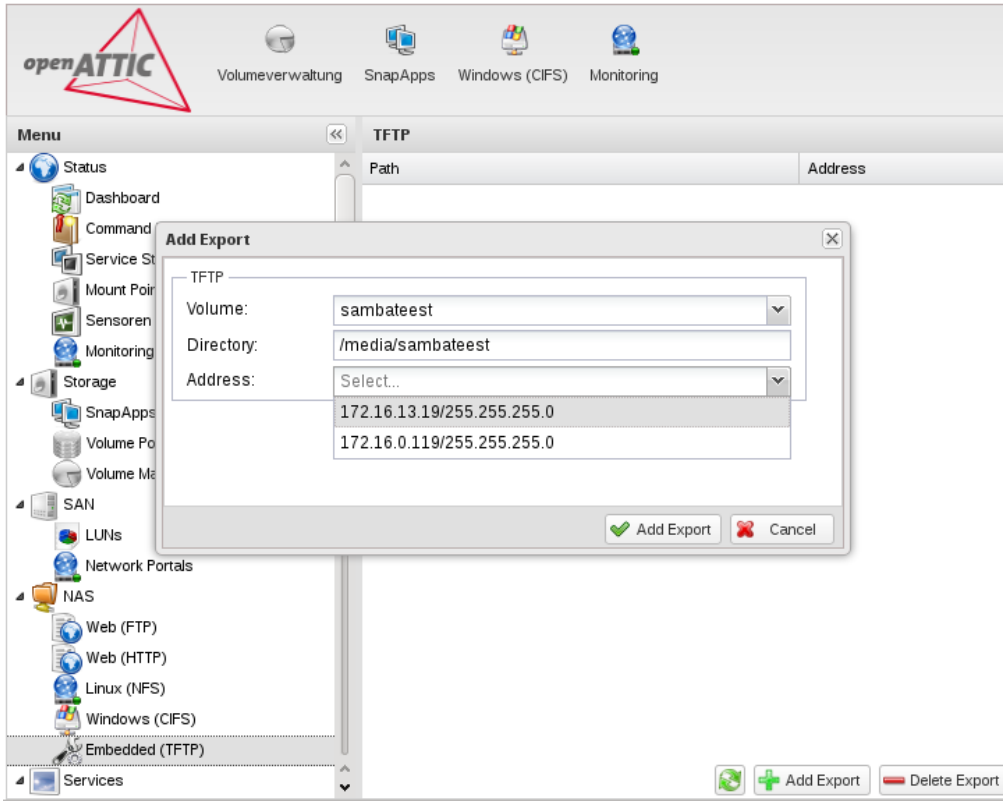
In contrast to HTTP mirrors, when using FTP you will usually want your users to be authenticated and permissions to be enforced, just like when using *CIFS* shares. In order to do that, openATTIC authenticates users via the Windows domain and enforces the same permissions. For technical reasons, currently all volumes on the system are exported – there is currently no way to create specific exports like with the other modules.

For this reason, this module does not offer any configuration options in the interface. However this may change in the future, when functionality is added to this module.

6.4.5 TFTP

When building embedded systems, the need for file transfer arises when it comes to keeping configuration files stored on a central server instead of on the devices themselves. However, implementing protocols like FTP on such devices may be challenging due to the severe limitations on resources. For these applications, Trivial File Transfer Protocol provides a remedy.

This protocol provides file transfer and file transfer *only*: There is no authentication, no resuming of failed downloads, no parallel downloads, just plain file transfer. Since there is also no way of selecting which volume to connect to, each IP address of the openATTIC node can only be used to export one single volume via TFTP.



To create a share, click the "Add Export" button, and provide the following information:

1. Choose the volume you would like to share.
2. If you would like to share only a certain subdirectory of the volume, you can enter its path in the `path` field. By default, this field is populated with the root directory of the volume.
3. Select the IP address on which the export is to be available.

Finally, click the `Add Export` button to create the new export.

6.5 Services

6.5.1 Cron Jobs

The cronjob panel lists all existing cronjobs and gives you detailed information about existing job - like the command which will be executed in the scheduled time, configuration information of the job and on which host this job runs.

You can add, edit and remove cron jobs as well.

In the background openATTIC uses the utility software cron, which is very useful in case you want to do something on a regular base or for a temporary period, for example taking a snapshot of a volume. By creating a job in the Cron Jobs panel openATTIC will create a crontab (cron table). Crontab is the configuration file, in which you can find the given information and the command you have scheduled in the openATTIC user interface.

6.5.2 Cron Syntax

You can use the following syntax

```
Minutes 0..59      | * for every minute
Hours 0..59        | * for every hour
Days 0..31         | * for every day
Month 1..12        | * for every month
Days of week 0..7 | * for every day of week
```

Note: 0 to 6 are Sunday to Sunday, 7 is like 0 also Sunday

You can use commas for separation.

6.5.3 Examples

Minutes	Hours	Day of month	Month	Day of week	Description
*	*	*	*	*	Every minute, every hour, seven days a week
10	*	*	*	*	Every day, 10 past midnight
10	0	*	*	3	Every Wednesday 10 past midnight
0	9	*	*	1	Every Monday morning at 09:00 a.m.

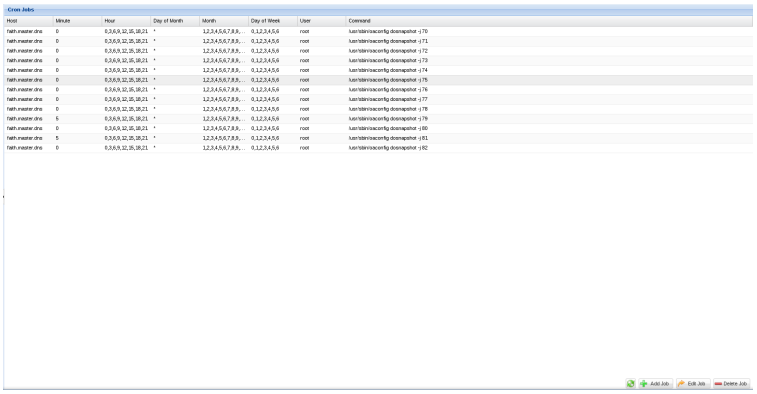
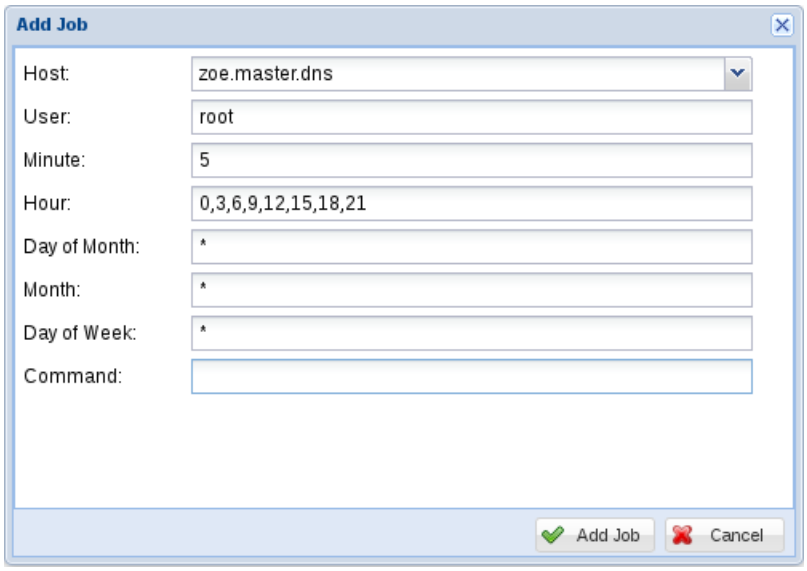


Figure 6.16: Cronjob overview

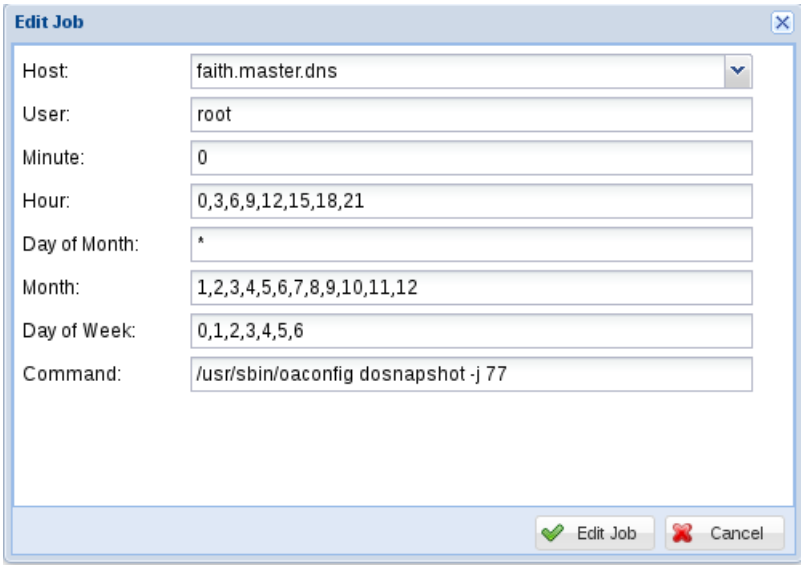
A more detailed description about how to add, edit and delete jobs can be found in the following text.

- **How to add a job:**
  - Hit the "Add Job" button
  - Select the host on which the script should be executed
  - Enter "root" as user
  - Schedule the time
  - Insert a command or path to a script

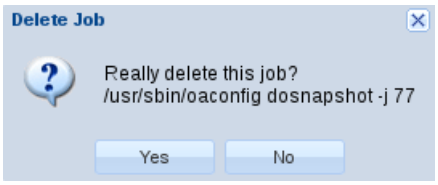
Note: You can enter the path where a script is located or for example use oaconfig like this: `/usr/sbin/oaconfig + command`.



- **This is where you can edit a job:**
  - Select the job you would like to edit and click the "Edit Job" button
  - Edit the data you want to change or replace
  - Click the "Edit Job" button



- **Delete job:**
  - Select the job you want to delete
  - Press the "Delete Job" button
  - Answer the confirmation message

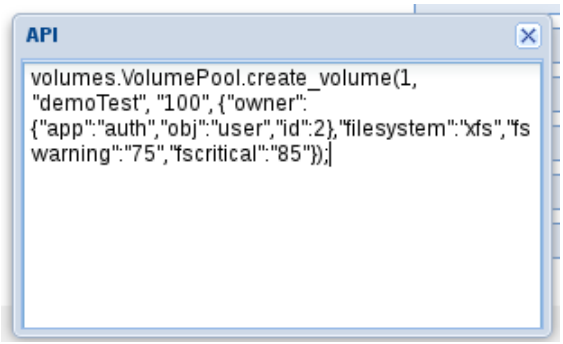


6.6 System

6.6.1 API record

By adding (for example) a volume in the openATTIC user interface the openATTIC backend (API) plays an important role. With the API record function, you can record those API-actions by clicking the menu entry on the left (subitem of "System") and it will start recording.

When you're done with recording actions in the user interface, click the API record entry again, then it will display the result in a separate window - which will look like this (add volume example):



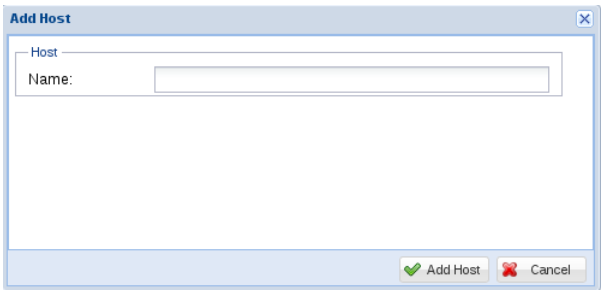
If you want to participate or develope your own plugin or extensions for openATTIC, this feature could be very useful, in order to see how openATTICs' API works, for example you can see the expected parameters of a function.

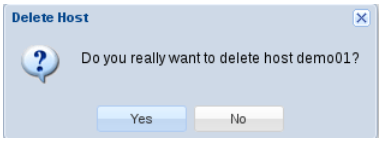
6.6.2 Hosts

Here you can see all hosts openATTIC is connected with. There are peers which are connected to openATTIC via an API-Key (for example to connect a cloud platform to openATTIC) as well as hosts which use storage provided by openATTIC and mapped via the protocols FibreChannel or iSCSI in order to extend the available storage of a virtualization host, for example.

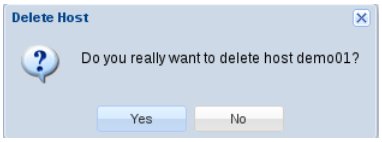
Hosts are listed on the left, on the right you can see the host attributes (peer, initiator). If you select a host, you will see a small arrow in front of the peer- and/or initiator directory if there are any entries.

- **Add host(s) in order to configure attributes for this host**
  - Enter the IP-address or hostname
  - click "Add Host" button
- **Edit host(s)**
  - Select the host you want to edit in the list
  - After editing the host click "Edit Host" button to confirm the changes
- **Delete Hosts(s)**





  - Select the host you want to delete in the list
  - Hit the "Delete Host" button
  - Answer the confirmation message



- **How to add an attribute to an host:**
  - Select the host and - depending on what you want to do - click on the peer or initiator folder and then click the "Add Attribute" button
  - **Add attribute form:**
    - Peer: Insert an API-Key in order to add a peer
    - Initiator: first select the protocol type (iSCSI or Fibre Channel), then insert a WWN/IQN for adding an initiator (see screenshots below)
  - click the "Submit" button.

  This is where you can add the peer attribute in order to create a connection between openATTIC and another host by creating an API-key:
- **Remove attribute from a host when you no longer need it:**
  - Select the host you want to delete an attribute from
  - Select the folder (peer, initiator) depending on which attribute you want to remove
  - Mark the attribute you want to delete and click the "Remove Attribute" button.
  - Answer the confirmation message to remove the attribute:

6.6.3 User management

You can manage openATTIC users and permissions here.

Here are the supported actions in the User Management panel:

- Add an openaTTIC user

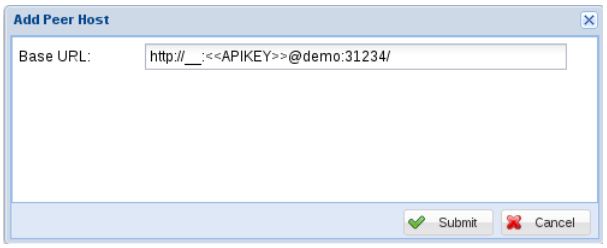
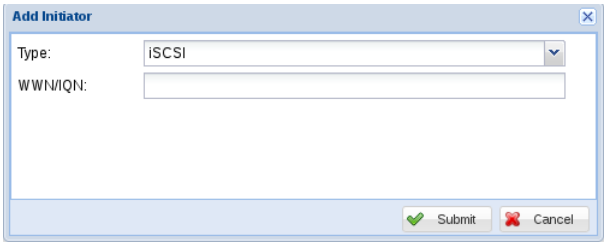


Figure 6.17: **This is where you can add the initiator attribute in order to map storage from openATTIC to a host in the list**



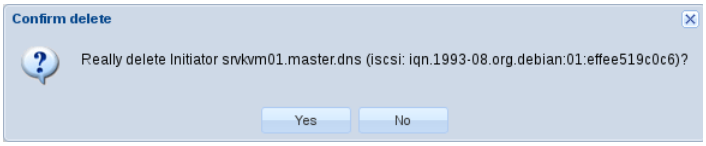
- Remove user accounts
- Change user permissions (deactivate / activate an account, set or unset SuperUser permissions)
- Change your password (see "edit existing user(s) / change password" below)
- register for monitoring notifications by adding an e-mail address to your account
- View the volumes of a user

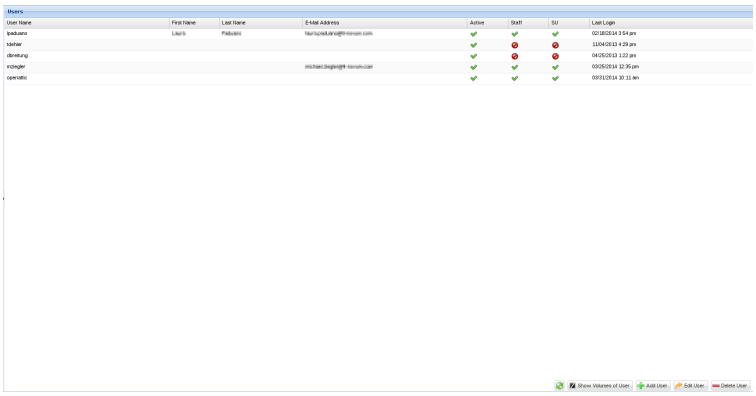
Creating different user accounts is useful, for example, when you want to add a volume and set a specific user as the owner. When creating a user account for a customer, you might want to create it without SuperUser permissions. It is also useful to create a user when you want to connect i.e. cloud platforms like openStack or openQRM to openATTIC. You need an API-Key to connect a cloud platform to openATTIC, this API-Key always belongs to a specific user. So you might want to do that with a special account.

A more detailed explanation can be found in the following text.

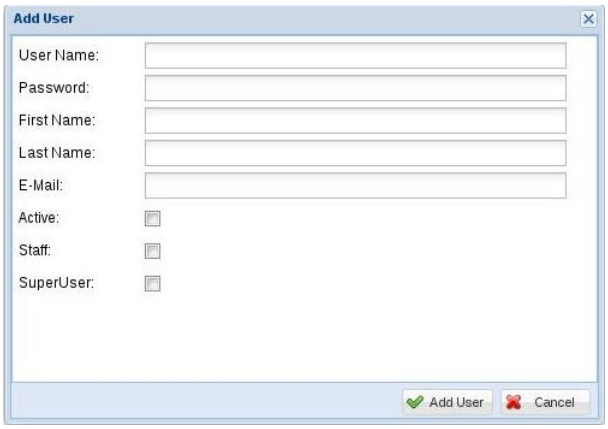
User management panel:

- **How to add a new openattic user:**
  - Click the "Add User" button
  - Enter a username and password
  - set user as active
  - choose permissions
  - **optional:**
    - First name
    - Last name





* E-Mail address (if you want to get notifications via Nagios)



*** This is where you can edit existing user(s) / change password**

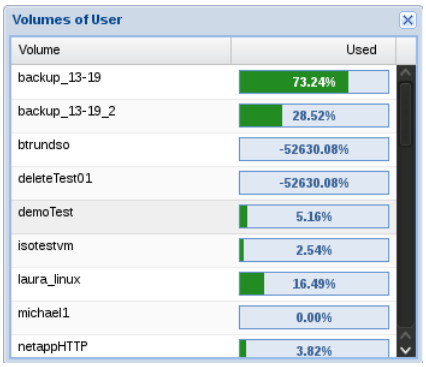
- Click the "Edit User" button
- Select the user you want to edit and click the "Edit User" button
- When you're done with editing the user information click the "Edit User" button in the form to save the changes
- **Delete user(s)**
  - Select the user you want to remove and click the "Delete User" button
- **show volumes where selected user is the owner**
  - select the user and click the "Show Volumes of User" button to see the list of volumes the chosen user owns:

6.6.4 API keys

API-Keys are required in order to create a trusted connection to an openATTIC host. After you've installed a fresh openATTIC system, you can find the API-Key "oacli access". This key ensures, that you can access the openattic commandline (oacli).

If you want to integrate openATTIC into another system, you'll need an API-Key. Otherwise, you can't connect to the openATTIC Host in order to request data or execute actions via the openATTIC API.

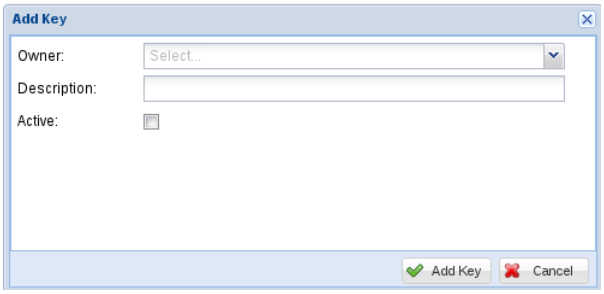
Supported actions of the API Keys panel:



- Add an API Key
- Edit an existing API Key
- Show API-Key-URL (in case you need the key, so you can copy and paste it)
- Remove an API Key

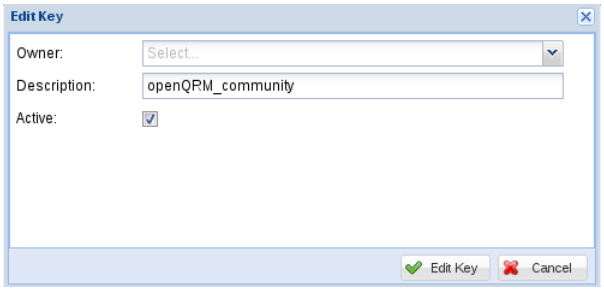
How to add an API-Key:

- click the "Add" button
- choose an openATTIC user as owner
- insert a description if needed
- make sure to set the key on "active"



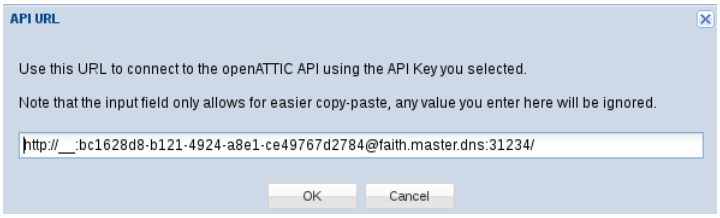
How to edit an API Key

- Select the API Key you want to edit in the list
- you can change the owner, the description or set the key to active or deactivate the key
- Click the "Edit Key" button to safe the changes



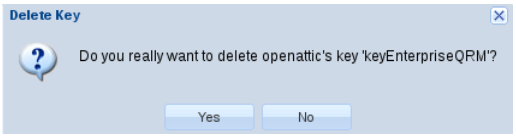
This is where you can view an API-Key-URL

- Select the API Key you want to see the URL
- click the “Show API URL” button
- click the “OK”- or “Cancel” button to close the window


API URL
Use this URL to connect to the openATTIC API using the API Key you selected.
Note that the input field only allows for easier copy-paste, any value you enter here will be ignored.
http://__:bc1628d8-b121-4924-a8e1-ce49767d2784@faith.master.dns:31234/
OK
Cancel

Delete an API Key

- Select the API Key you want to delete
- answer the confirmation message

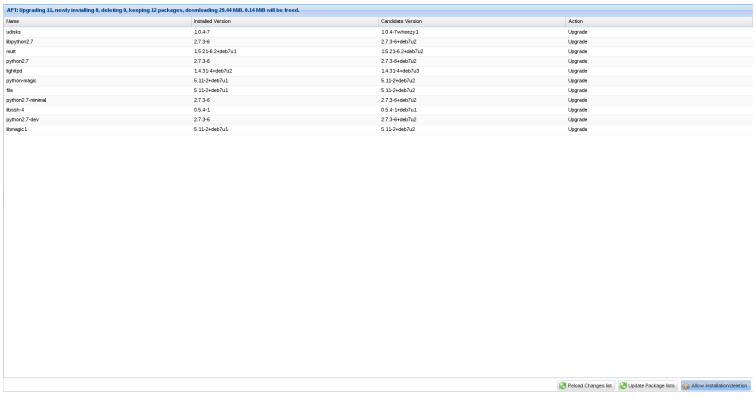


6.6.5 Online update

In case there are some packages which need to be updated, you will see them in the Online Update panel. There are also detailed information about the package name, the installed version, the candidate version and which action (install, upgrade, delete) is necessary to get the latest version. Some packages are no longer needed, so they will be removed.

Available options:

- Reload Changes list
- Update Package list
- Allow installation/deletion



6.7 Personal Settings

The Personal Settings panel summarizes some settings in order to give you one central panel where you can configure and adapt openATTICs' user interface to your personal needs:

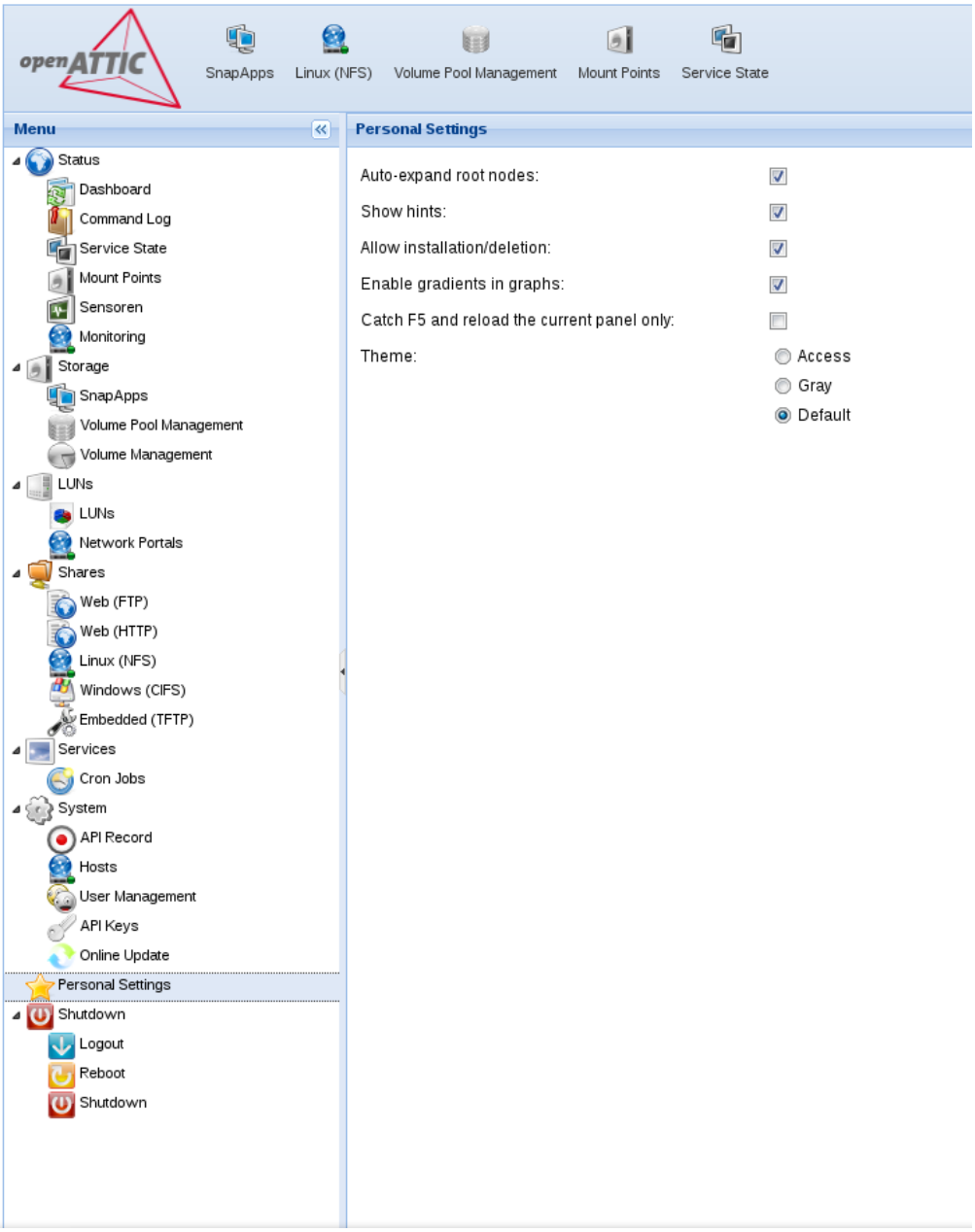


Figure 6.18: Personal Settings panel

Note: By checking or unchecking an option openATTIC will save the configuration - you just have to reload the page, except when changing the theme.

Let's have a closer look at the offered options:

Auto-expand root nodes (enabled by default)

- view all menu entries and their subitems
- to see the menu entries without their subitems uncheck it

Show hints (enabled by default)

- default is checked: gives you configuration options/hints for different cases if needed, for example in the volume management panel when adding a volume:

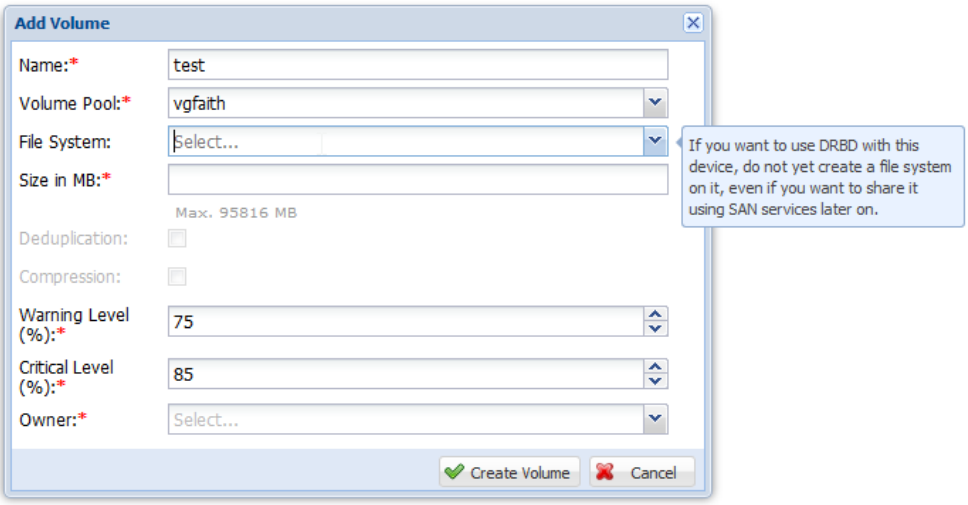


Figure 6.19: hints activated

Allow installation/deletion (enabled by default):

- Allow openATTIC to automatically install required packages or remove unnecessary packages

Enable gradients in graph (not enabled by default):

- Displays graph with gradients in the monitoring panel and graphs which were added to the dashboard site

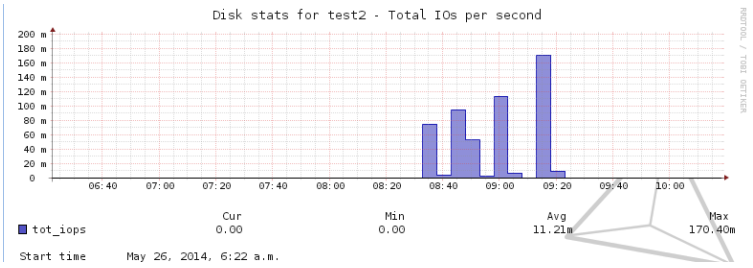


Figure 6.20: This is how graphs look like without gradient

Catch F5 and reload the current panel only (not enabled by default):

- won't reload the whole page but the current viewed pannel

Theme:

- you can switch between the designs "Access", "Gray" and "Default"
- just select the radio button of the theme you would like to configuare as your standard theme openATTIC will reload the page with the selected theme then

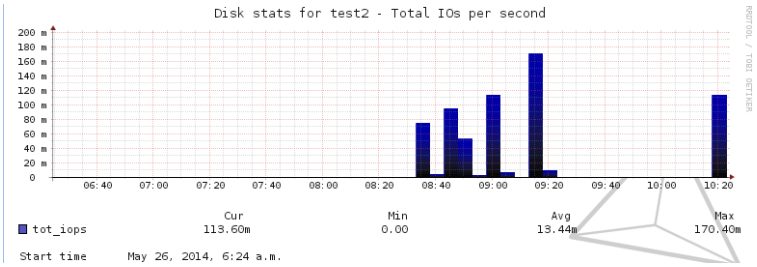


Figure 6.21: Graphs with gradient

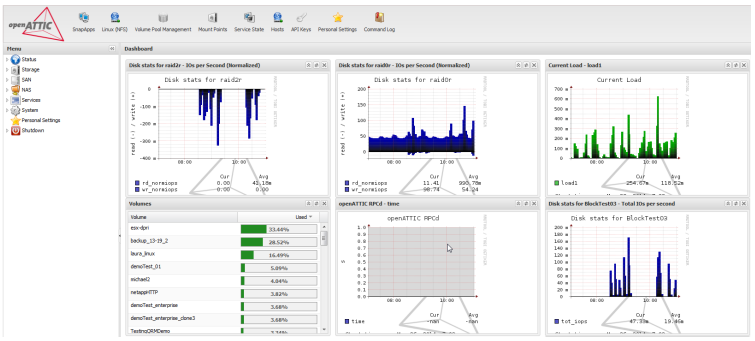


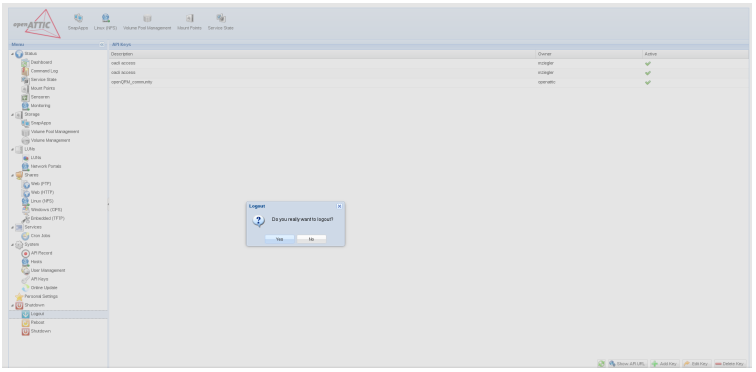
Figure 6.22: openATTIC with theme “Gray”, graph with gradients and auto-expand root nodes unchecked

6.8 Shutdown

This is where you can exit the openATTIC user interface, shutdown or reboot the openATTIC server.

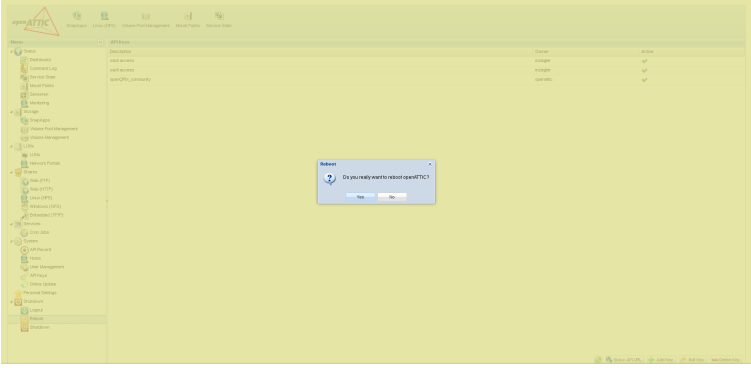
6.8.1 Logout

Leave the openATTIC user interface by clicking the “Logout” button and answering the confirmation message.



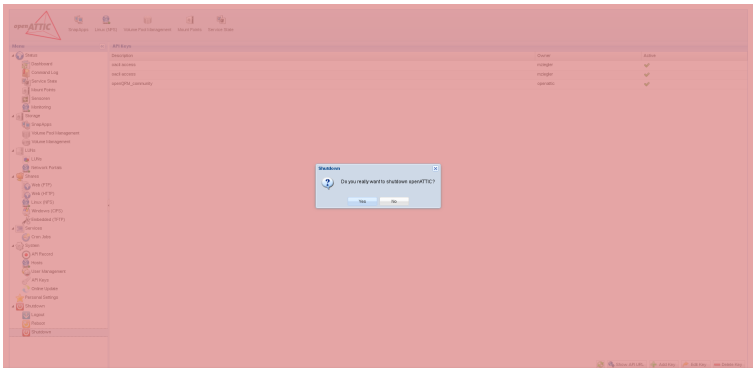
6.8.2 Reboot

Reboot the openATTIC server by clicking the “Reboot” button and answering the confirmation message.



6.8.3 Shutdown

Shutdown the openATTIC server by clicking the "Shutdown" button and answering the confirmation message.



6.9 Menu shortcut bar

How to add a shortcut bar:

You can add subitems of Status, Storage, SAN (...) of the menu tree on the left into the menu shortcut bar (space next to the openATTIC logo).

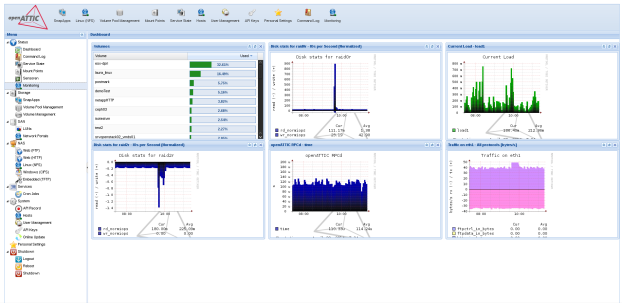


Figure 6.23: Just select the item and add it via drag and drop

How to remove a shortcut bar:

To remove the subitem from the menu shortcut bar select the item and drag it to the end of the shortcut bar, drop the item and answer the security question.

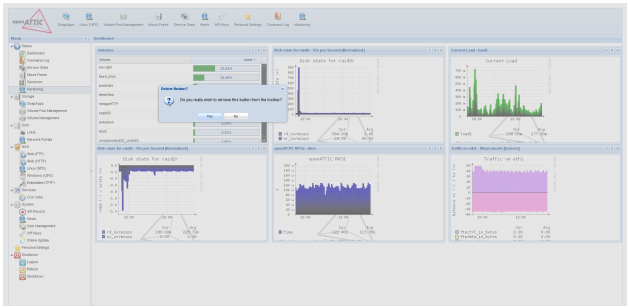
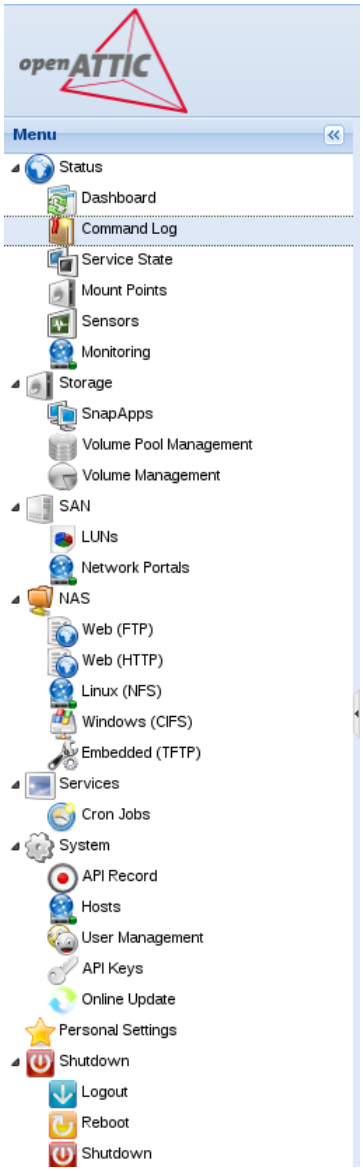


Figure 6.24: remove shortcut bar

6.10 Hiding the menu tree

By clicking the "<<" button in (upper right-hand corner) of the menu tree-panel or the small arrow (middle of the menu panel on the right) you can hide the whole menu. Click the same button to view the menu again.


openATTIC
Menu
Status
Dashboard
Command Log
Service State
Mount Points
Sensors
Monitoring
Storage
SnapApps
Volume Pool Management
Volume Management
SAN
LUNs
Network Portals
NAS
Web (FTP)
Web (HTTP)
Linux (NFS)
Windows (CIFS)
Embedded (TFTP)
Services
Cron Jobs
System
API Record
Hosts
User Management
API Keys
Online Update
Personal Settings
Shutdown
Logout
Reboot
Shutdown

CHAPTER

SEVEN

INTEGRATION

System requirements are subject to change on a daily basis: New services are installed all the time, obsoleting others, and the storage system has to adapt to these changes. Hence, being able to integrate openATTIC into other processes is important.

openATTIC provides an XML-RPC API which allows other infrastructure parts to access its functionality. This way, processes can be easily automated, ensuring that processes run the way they are supposed to.

The *Cloud Storage* use case outlines different products which are supported in conjunction with openATTIC. The following section focuses on the installation of the respective cloud connectors. In case such a connector .. derek: <s>respective</s> does not exist for the software you want to use, the *Integration Tutorial* tells you everything you need to get started building your own cloud connector.

7.1 Cloud Connectors

Cloud systems commoditize infrastructure, meaning that an end user can order virtual servers, storage volumes and network segments to be configured a la carte. The cloud system completely automates the creation and management processes that are necessary in order to deliver whatever the customer ordered.

A translation between the cloud system's inner workings and openATTIC's API is necessary. The cloud connectors handle this translation, enabling the cloud system to do its job.

7.1.1 OpenStack

Note: We are currently working on this part of our documentation. An update will be available soon.

7.1.2 openQRM

Note: We are currently working on this part of our documentation. An update will be available soon.

7.2 XML-RPC API

The openATTIC API is the key component when it comes to automating storage management processes. It is available on port 31234 and accessible using standard XML-RPC. This section outlines the necessary administrative preparations that need to be made and shows how to use the API programmatically, giving code examples in Python and PHP.

7.2.1 Authentication

API calls need to be authenticated by using standard HTTP Basic Authentication. openATTIC accepts two forms of credentials:

1. Username and password of an administrator account.
2. __ (two underscores) as the username and an API key as the password.

Using API keys is recommended, because keys provide a means of authentication that does not break when the user changes their password and that can be revoked individually without affecting other applications.

API keys can be managed in the *API keys* GUI panel.

7.2.2 Quick Start Example

In Python, accessing the API using username and password works like this:

```
>>> from xmlrpclib import ServerProxy
>>> sp = ServerProxy("http://<user>:<password>@<host>:31234/")
>>> sp.volumes.StorageObject.all()
```

Or, using an API key:

```
>>> from xmlrpclib import ServerProxy
>>> sp = ServerProxy("http://__:<apikey>@<host>:31234/")
>>> sp.volumes.FileSystemVolume.filter({"pool": 5})
```

Warning: Python's xmlrpclib is not thread safe, so be careful when writing threaded applications. In particular, do not share ServerProxy instances between multiple threads.

The following works for PHP, when using the openATTIC XMLRPC Proxy Library which takes care of the encoding busywork:

```
<?php
include("openattic-client.php");
$oa = new OpenAtticProxy("http://__:<apikey>@<host>:31234/");
print_r($oa->volumes->StorageObject->all());
print_r($oa->volumes->FileSystemVolume->filter(array("pool" => 5)));
```

7.2.3 Deep dive

The examples above gave you a very quick overview on how to connect to the API. Of course, the API provides a rich set of functions, which can be used to control every part of the openATTIC system. In fact, the GUI uses it itself, so there is no point in the GUI that is not accessible via the API.

For easy exploration and testing, every openATTIC system comes with a command-line utility named *oacli*, which shows you which sections and functions are available and what their parameters are.

The same information can be found in the *Available functions* section.

openATTIC API Client: `oacli`

openATTIC comes with a tool which you can use to access the RPC-API. This tool is very useful in order to execute tests or work with openATTIC via commandline instead of using the graphical interface.

Sections

By starting oacli it creates its structure and sections based on the information it gets from the API. Word completion can be used by pressing Tab-key:

```
$ oacli
srvopenqrmsto01:#>
auth                fqdn                help                lvm                 pkgapt              t
clustering          ftp                 hostname            munin               rpcd                u
cmdlog              get_function_args   hoststats           nagios              samba
drbd                get_installed_apps  http                nfs                 shell
end                 get_loaded_modules  ifconfig            peering             system
exit                get_object          iscsi               ping                sysutils
srvopenqrmsto01:#> lvm
srvopenqrmsto01:lvm>
LogicalVolume  VolumeGroup    ZfsSnapshot    ZfsSubvolume   end            exit           help
srvopenqrmsto01:lvm> LogicalVolume
srvopenqrmsto01:lvm.LogicalVolume>
```

The commandline shows the hostname of the host the Shell is currently connected to as well as the section you are in. The root section is marked by a hash (`#`).

You can exit a section by typing two dots (`..`) and pressing enter.

Exit the oacli by typing `exit` or using `ctrl+D`.

Help

You can find a help-command in every section, which lists existing commands as well as a short documentation of the command within the section:

```
srvopenqrmsto01:lvm.LogicalVolume> help

Documented commands (type help <topic>):
========================================
all         end           filter_values  help           lvm_info   set_ext
all_values  exit          fs_info        idobj          mount      unmount
avail_fs    filter        get            ids            mount_all  unmount_all
create      filter_combo  get_ext        is_in_standby  remove
disk_stats  filter_range  get_shares     is_mounted     set



Miscellaneous help topics:
==========================
syntax  sections

srvopenqrmsto01:lvm.LogicalVolume> help filter
Usage: lvm.LogicalVolume.filter <kwds>

Search for objects with the keywords specified in the kwds dict.

            'kwds' may contain the following special fields:

            * __exclude__: ''**kwargs'' for an .exclude() call.
            * __fields__: ''*args'' for a .values() call.
```

```
            Any other fields will be passed as ``**kwargs`` to .filter().
            See the `Django docs <https://docs.djangoproject.com/en/dev/topics/db/queries/>`_ for det
```

The commands `help syntax` and `help sections` do not refer to the commands themselves, they only give a short overview of the introduction of the structure of the oacli-shell.

Call-Up

The existing commands are identic in every section with the exported methods within those sections. Some of the methods expect complex arguments, which consists of objects. For such cases oacli supports simple arguments as you would do it in a bash as well as the input of an object in JSON-syntax:

```
srvopenqrmsto01:lvm.LogicalVolume> get 15
{
    ...
    "name": "mirror_debian_squeeze",
    ...
}
srvopenqrmsto01:lvm.LogicalVolume> filter {"name__icontains": "deb"}
[
    {
        ...
        "name": "debpkgtest",
        ...
    },
    {
        ...
        "name": "mirror_debian_squeeze",
        ...
    }
]
```

Shell-Section

In order to change options of the shell at run time you can use the special section called "shell" which is not part of the API. This section offers among other things the possibility to change the output format as well as list the history of entered commands or to delete the history.

Available functions

This part documents the available functions by modules.

Note: Every module includes all the methods defined in `rpcd.handlers.ModelHandler`, so this section actually covers most of what you're going to need.

Note: In order to operate on volumes, you should always refer to the `volumes.rpcapi.StorageObjectHandler` and `volumes.rpcapi.StorageObjectProxy` classes instead of using the concrete volume modules (e.g., lvm) directly. That way, your code will just work with whatever storage backend openATTIC is configured to use.

Base Handlers

7.3 Integration Tutorial

Learning by example is always easier than wolfing through endless pages of documentation, so let's have a quick look at the openATTIC API by writing a short program that actually uses it.

We are going to write a shell script that automatically provisions virtual machines using libvirt and KVM, storing VM images in openATTIC volumes.

7.3.1 Prerequisites

We'll assume that you have two systems ready: One is a blank Debian or Ubuntu installation, the other one a basic openATTIC installation. The systems should have enough resources that running VMs on them is any fun, but apart from that, they don't have to be anything fancy for now.

Being a bit familiar with Python also helps, but the scripts we're going to use are pretty short and easy to understand.

If you want to see the actual VM deployment process in action, you should also have a QCOW2 VM image handy, from which the VMs will be cloned.

7.3.2 First steps in oacli

First of all, we're going to make sure the RPC API is available by creating a volume using *oacli*. Connect to the openATTIC system via `ssh` and run the `oacli` command. It will greet you with a shell prompt such as this:

```
root@faith:~$ oacli
Initialized config from /etc/openattic/cli.conf
faith:#>
```

Now you're ready to interact with the openATTIC API directly, so let's find out what it can do.

Listing volume pools

First of all, we'll take a look at available volume pools to find one to use for our volumes:

```
faith:#> volumes
faith:volumes> StorageObject
faith:volumes.StorageObject> ids_filter {"volumepool__isnull": false}
[
   {
      "obj": "StorageObject",
      "app": "volumes",
      "filesystemvolume": {
            "app": "btrfs",
            "obj": "BtrfsSubvolume",
            "id": 633,
            "__unicode__": "btrtest"
      },
      "id": 24,
      "blockvolume": {
            "app": "lvm",
            "obj": "LogicalVolume",
            "id": 479,
```

```
            "__unicode__": "btrtest"
        },
        "volumepool": {
            "app": "btrfs",
            "obj": "Btrfs",
            "id": 6,
            "__unicode__": "btrtest"
        },
        "__unicode__": "btrtest"
    },
    {
        "obj": "StorageObject",
        "app": "volumes",
        "filesystemvolume": {
            "app": "zfs",
            "obj": "Zfs",
            "id": 581,
            "__unicode__": "tank"
        },
        "id": 45,
        "blockvolume": null,
        "volumepool": {
            "app": "zfs",
            "obj": "Zpool",
            "id": 5,
            "__unicode__": "tank"
        },
        "__unicode__": "tank"
    },
    {
        "obj": "StorageObject",
        "app": "volumes",
        "filesystemvolume": null,
        "id": 43,
        "blockvolume": null,
        "volumepool": {
            "app": "lvm",
            "obj": "VolumeGroup",
            "id": 1,
            "__unicode__": "vgfaithdata"
        },
        "__unicode__": "vgfaithdata"
    }
]
```


So, our openATTIC system named “faith” knows three volume pools: A LVM volume group named “vgfaithdata”, a Zpool named “tank”, and a btrfs volume named “btrtest”. Which one of these we use is completely up to us: The code we’re going to write doesn’t care one bit. Since we have to choose one, we’ll use the Zpool, because it has an SSD cache and the RAID where the VG and the btrfs reside upon does not have a BBU – so, using the Zpool should give us a lower storage latency.

Note: Which pool to use is completely hardware-dependent. The zpool being the better choice in *this* case does **not** imply that ZFS is the best choice in *all* cases.

Understanding StorageObjects

If you take a closer look at the objects returned by the API, you will notice that openATTIC refers to them as `StorageObjects`:


```
{
   "obj": "StorageObject",
   "app": "volumes",
   "filesystemvolume": {
         "app": "btrfs",
         "obj": "BtrfsSubvolume",
         "id": 633,
         "__unicode__": "btrtest"
   },
   "id": 24,
   "blockvolume": {
         "app": "lvm",
         "obj": "LogicalVolume",
         "id": 479,
         "__unicode__": "btrtest"
   },
   "volumepool": {
         "app": "btrfs",
         "obj": "Btrfs",
         "id": 6,
         "__unicode__": "btrtest"
   },
   "__unicode__": "btrtest"
}
```


In the storage world, multiple concepts exist to describe storage, which may or may not apply to the same thing, depending on what the thing actually is and how it has been configured. Taking a closer look at the btrfs volume pool above, you can see that this object uses all three of openATTIC's high-level abstractions:

1. The `filesystemvolume` part indicates, that this object provides a file-system somewhere. Hence, it can be shared using NAS protocols and accessed using the likes of Windows Explorer.
2. The `blockvolume` part means that this object is a block device. This is because this btrfs file system has been created inside an LVM logical volume, which can be formatted with a file system (like we have done here) or shared via SAN protocols, to let the client handle the formatting in whatever way they like.
3. Finally, the `volumepool` part is to say that this thing supports the creation of subvolumes.

The fact that this StorageObject provides all three abstractions is fortunate because it makes this demonstration a bit easier, but does not *have* to be the case. For example, an LVM Volume Group will provide the `volumepool` information *only*, and since the "tank" zpool from the above output has not been created inside an LVM logical volume, its `blockvolume` part is empty as well.

Creating a volume

Let's see how creating a volume in this pool works in oacli:

```
faith:volumes.StorageObject> help create_volume
Usage: volumes.StorageObject.create_volume <id> <name> <megs> <options>

Create a volume in this pool.

Options include:
```

```
 * filesystem: The filesystem the volume is supposed to have (if any).
 * owner:      The owner of the file system.
 * fswarning:  Warning Threshold for Nagios checks.
 * fscritical: Critical Threshold for Nagios checks.

What exactly this means is up to the volume implementation.
```

Note: The `help` command is implemented everywhere and documents all the available commands and their parameters.

The `create_volume` method expects four arguments: The volume pool id, volume name and size, and a couple of volume-dependent options. For ZFS, we'll need to set all of them, like so:

```
faith:volumes.StorageObject> create_volume 45 tutorial_vm01 1000 {"filesystem": "zfs", "owner": {"app
{
   "obj": "StorageObject",
   "app": "volumes",
   "filesystemvolume": {
      "app": "zfs",
      "obj": "Zfs",
      "id": 650,
      "__unicode__": "tank/tutorial_vm01"
   },
   "id": 127,
   "blockvolume": null,
   "volumepool": null,
   "__unicode__": "tutorial_vm01"
}
```

These parameters have the following meaning.

- `45`: The id of the volume pool in which the volume is supposed to be created.
- `tutorial_vm01`: The name of the volume.
- `1000`: The size of the volume in Mebibytes.
- `filesystem: zfs`: The filesystem to use for the new volume.
- `fswarning: 95`: Volumes that contain VM images don't degrade in performance when filling up, so a high warning threshold is ok.
- `fscritical: 98`: Same goes for critical.
- `owner: {"app": "auth", "obj": "User", "id": 2}` The new volume owner's user ID.

The `owner` field will have to be explained in a little more detail. What you're passing there is a reference to another object in the openATTIC system, in this case a user. These objects can be acquired by looking up which users exist:

```
faith:#> auth
faith:auth> User
faith:auth.User> ids
[
   {
      "app": "auth",
      "obj": "User",
      "id": 2,
      "__unicode__": "mziegler"
   },
   {
```

```
      "app": "auth",
      "obj": "User",
      "id": 1,
      "__unicode__": "openattic"
   }
]
```

So, the user ID `2` refers to the system user `mziegler`.

Exporting the volume

Now that we created a subvolume, we need to export it on the openATTIC system in order to be able to use it. Hence, we'll create an NFS export for it:

```
faith:#> nfs
faith:nfs> Export
faith:nfs.Export> create {"volume": {"app": "volumes", "obj": "FileSystemVolume", "id": 650}, "addres
{
   "app": "nfs",
   "obj": "Export",
   "id": 37,
   "__unicode__": "tutorial_vm01 - 172.16.13.132"
}
```

Again, here's what the parameters mean:

- `volume`: The FileSystemVolume ID of the volume we just created. NFS is a NAS protocol and therefore requires a file system to exist on the volume, which is why only FileSystemVolume can be specified here.
- `address`: The address for the *target node* which is going to be allowed to access the volume.
- `options`: NFS options for the share. The options listed here are the defaults, which usually work fine for VMs.
- `path`: If we wanted to share only a subdirectory of the volume, we could specify that here. We don't want to, so we specify the root path of the volume.

Mounting the volume

Now that we have created and exported a volume, let's see if this worked and verify the target system, "zoe", is allowed to see it:

```
root@zoe:~$ showmount -e faith
Export list for faith:
/media/tank/tutorial_vm01                                    172.16.13.132
```

There we go – we're now able to mount it somewhere:

```
root@zoe:~$ mkdir /media/tutorial_vm01
root@zoe:~$ mount -overs=3 faith:/media/tank/tutorial_vm01 /media/tutorial_vm01
root@zoe:~$ df -h
Filesystem                          Size  Used Avail Use% Mounted on
rootfs                              130G    5,2G  118G    5% /
udev                                 10M       0   10M    0% /dev
tmpfs                               1,2G    304K  1,2G    1% /run
tmpfs                               5,0M       0  5,0M    0% /run/lock
tmpfs                               2,4G       0  2,4G    0% /run/shm
faith:/media/tank/tutorial_vm01     737G       0  737G    0% /media/tutorial_vm01
```

Hooray, we have successfully mounted the volume!

7.3.3 Automating the export process

Now that we know the steps we're going to have to take, we can of course automate them using a Python script – but first of all, we're going to need an API key in order to access it. We can easily create one using the *API keys* GUI panel:

1. In the menu tree, navigate to the "API keys" panel.
2. Klick the "Add key" button.
3. Select an owner for the key, and give it a useful description so that you can easily identify it later on. Be sure to check the "Active" check box, otherwise the key won't work.
4. Submit the form via the "Add key" button. The form will disappear, and a new key will be created.
5. Display the URL by right-clicking the key and choosing "Show API URL" from the menu. It will look something like this: `http://__:eaa8eff0-bc93-45d1-beb2-ac61a8748e84@faith.master.dns:31234/`

Copy-paste the URL somewhere, you'll need it in a bit.

Putting an API client together

A python script that automates the volume creation and export process may look like this:

```
import sys

from xmlrpclib import ServerProxy

if len(sys.argv) < 2:
      print "Usage: python createvolume.py [vm name]"

vmname  = sys.argv[1]

# Paste your API URL here:
oa = ServerProxy("http://__:eaa8eff0-bc93-45d1-beb2-ac61a8748e84@faith:31234/")

volume_id = oa.volumes.StorageObject.create_volume(45, vmname, 1000, {
   "filesystem": "zfs",
   "fswarning":  95,
   "fscritical": 98,
   "owner": {"app": "auth", "obj": "User", "id": 2}
})

oa.volumes.StorageObject.wait(volume_id["id"], 600)

oa.nfs.Export.create({
   "volume":  volume_id["filesystemvolume"],
   "path":    "/media/tank/" + vmname,
   "options": "rw,no_subtree_check,no_root_squash",
   "address": "172.16.13.132"
})
```

The sections explained:

1. Loading libraries and parsing arguments:

```
import sys

from xmlrpclib import ServerProxy

if len(sys.argv) < 2:
      print "Usage: python createvolume.py [vm name]"

vmname  = sys.argv[1]
```

2. Initializing the connection to openATTIC:

```
oa = ServerProxy("http://__:eaa8eff0-bc93-45d1-beb2-ac61a8748e84@faith:31234/")
```

> **Warning:** Python's xmlrpclib is not thread safe, so be careful when writing threaded applications. In particular, do not share ServerProxy instances between multiple threads.

3. Creating the volume the same way we did in `oacli`:

```
volume_id = oa.volumes.StorageObject.create_volume(45, vmname, 1000, {
   "filesystem": "zfs",
   "fswarning":  95,
   "fscritical": 98,
   "owner": {"app": "auth", "obj": "User", "id": 2}
})
```

4. Waiting for the volume creation process to complete:

```
oa.volumes.StorageObject.wait(volume_id["id"], 600)
```

5. Creating the NFS export for the new volume, the same way we did in `oacli`:

```
oa.nfs.Export.create({
   "volume":  volume_id["filesystemvolume"],
   "path":    "/media/tank/" + vmname,
   "options": "rw,no_subtree_check,no_root_squash",
   "address": "172.16.13.132"
})
```

Of course, this script hardcodes way too much in order to be usable in a production system, but it'll do for our first steps.

Copy-paste the script into a file named `createvolume.py`, and be sure to replace the API url with your own. Now let's see if it works:

```
root@zoe:~$ python createvolume.py tutorial_vm02
root@zoe:~$ showmount -e faith
Export list for faith:
/media/tank/tutorial_vm02                                  172.16.13.132
/media/tank/tutorial_vm01                                  172.16.13.132
root@zoe:~$ mkdir /media/tutorial_vm02
root@zoe:~$ mount -overs=3 faith:/media/tank/tutorial_vm02 /media/tutorial_vm02
root@zoe:~$ df -h
Filesystem                          Size  Used Avail Use% Mounted on
...
faith:/media/tank/tutorial_vm01     737G  1,0M  737G   1% /media/tutorial_vm01
faith:/media/tank/tutorial_vm02     737G  1,0M  737G   1% /media/tutorial_vm02
```

This looks promising. openATTIC created a new volume named `tutorial_vm02`, exported it via NFS, and we were able to mount it successfully.

Automating the mount process

Now that we're able to create and export volumes, let's automate the mounting part. We'll use a shell script for this, because this part is not as easy to implement in Python. Put the following in a file named `createvm.sh`:

```
#!/bin/bash

if [ -z "$1" ]; then
      echo "Usage: $0 <vm name>"
      exit 1
fi

VM="$1"

python createvolume.py "$VM"
mkdir /media/$VM
mount -overs=3 faith:/media/tank/$VM /media/$VM
echo faith:/media/tank/$VM /media/$VM nfs vers=3,auto 0 0 >> /etc/fstab
```

Now, let's run it a few times and see if it works:

```
root@zoe:~$ ./createvm.sh tutorial_vm03
root@zoe:~$ ./createvm.sh tutorial_vm04
root@zoe:~$ ./createvm.sh tutorial_vm05
root@zoe:~$ df -h
Filesystem                          Size  Used Avail Use% Mounted on
...
faith:/media/tank/tutorial_vm01     737G  1,0M  737G   1% /media/tutorial_vm01
faith:/media/tank/tutorial_vm02     737G  1,0M  737G   1% /media/tutorial_vm02
faith:/media/tank/tutorial_vm03     737G  1,0M  737G   1% /media/tutorial_vm03
faith:/media/tank/tutorial_vm04     737G  1,0M  737G   1% /media/tutorial_vm04
faith:/media/tank/tutorial_vm05     737G  1,0M  737G   1% /media/tutorial_vm05
```

Looks great so far. Now, let's create some actual virtual machines in these volumes.

7.3.4 Automatic VM deployment

In the introduction, we noted you should also have a QCOW2 VM image to clone deployed VMs from. First, copy this image to `/media/base.qcow2`.

Extending `createvm.sh`

Now, in order to run VMs from it, we will create new QCOW2 images that are based upon the original image and store all the changes, so that VMs will boot from the original image, but will not modify it so it can be used over and over again.

The command to do this looks like this:

```
qemu-img create -f qcow2 -b /media/base.qcow2 /media/$VM/hda.qcow2 10G
chown -R libvirt-qemu:libvirt-qemu /media/$VM
```

Append these commands to the end of `createvm.sh`.

Next, we'll add code that creates a configuration file for `libvirt` and defines the new VM from it. The new VM will have

- 1GiB of RAM,

- 1 CPU,
- the boot disk set to the newly created image,
- a random VNC port.

The code to create the VM is a bit lengthy:

```
TEMPFILE="`tempfile`"
cat > "$TEMPFILE" <<EOF
<domain type='kvm'>
<name>$VM</name>
<memory unit='KiB'>1048576</memory>
<currentMemory unit='KiB'>1048576</currentMemory>
<vcpu placement='static'>1</vcpu>
<os>
   <type arch='x86_64' machine='pc-1.0'>hvm</type>
   <boot dev='hd'/>
</os>
<features>
   <acpi/>
   <apic/>
   <pae/>
</features>
<clock offset='utc'/>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>restart</on_crash>
<devices>
   <emulator>/usr/bin/kvm</emulator>
   <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2' cache='none' aio='threads'/>
      <source file='/media/$VM/hda.qcow2'/>
      <target dev='hda' bus='virtio'/>
   </disk>
   <interface type='bridge'>
      <source bridge='br0'/>
      <model type='virtio'/>
   </interface>
   <serial type='pty'>
      <target port='0'/>
   </serial>
   <console type='pty'>
      <target type='serial' port='0'/>
   </console>
   <input type='tablet' bus='usb'/>
   <input type='mouse' bus='ps2'/>
   <graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0' keymap='en-us'>
      <listen type='address' address='0.0.0.0'/>
   </graphics>
   <video>
      <model type='cirrus' vram='9216' heads='1'/>
   </video>
   <memballoon model='virtio'>
   </memballoon>
</devices>
</domain>
EOF

virsh define "$TEMPFILE"
```

```
rm -f "$TEMPFILE"
```

Lastly, to start the VM, add:

```
virsh start $VM
```

Testing it

Running the script should now produce the following output:

```
root@zoe:~$ ./createvm.sh tutorial_vm06
Formatting '/media/tutorial_vm06/hda.qcow2', fmt=qcow2 size=10737418240 backing_file='/media/base.qc
Domain tutorial_vm06 defined from /tmp/fileZH42zZ

Domain tutorial_vm06 started
```

Let's see if that's actually true by checking `libvirt`:

```
root@zoe:~$ virsh list
Id    Name                           State
----------------------------------------------------
1     tutorial_vm06                  running
```

In order to connect to the VM, find out the VNC display:

```
root@zoe:~$ virsh vncdisplay tutorial_vm06
:0
```

So, connecting to the host's IP address and the given VNC port should now work, and show you a shiny new virtual machine.

7.3.5 Congratulations!

You have just implemented an automated virtual machine deployment process. Note that the part that deals with the actual openATTIC API is the little Python script from the *Putting an API client together* chapter – all the crufty stuff is on the hypervisor side, which will usually be handled by some kind of virtualization management system. Check out *Cloud Storage* for more information.

CHAPTER

EIGHT

DEVELOPER DOCUMENTATION

If you want to help, you don't have to be a coder: reporting bugs also helps!

openATTIC consists of a set of components built on different frameworks, which work together to provide a comprehensive storage management platform.

When an Application, be it OpenStack or the oA GUI, wants stuff to be done, this is what happens:

- The RPC-API receives a request in form of a function call, decides which host is responsible for answering the request, and forwards it to the core on that host.
- The openATTIC Core consists of two layers:
  - Django Models, the brains. They keep an eye on the whole system and decide what needs to be done.
  - FileSystem layer: Decides which programs need to be called in order to implement the actions requested by the models, and calls those programs via Systemd.
- The Systemd executes commands in the system and delivers the results.

First of all, start off by *Setting up a development system*. Then code away, implementing whatever changes you want to make. When you're done, decide whether or not you want to *submit the code*.

8.1 Setting up a development system

In order to begin coding on openATTIC, you will require a development system. Setting one up can be easily done with the following steps.

1. openATTIC requires a bunch of tools and software to be installed and configured, which is handled automatically by the Debian packages. While you could of course configure these things manually, doing so would involve a lot of manual work which isn't really necessary. Set up the system just as described in *Installation and Upgrade Guides*, but **do not yet execute ''oaconfig install''**. We recommend using the Nightly Build for dev systems.
2. Set the installed packages on `hold` to prevent Apt from updating them.
3. Install Mercurial.
4. Go to the `/srv` directory, and clone the openATTIC repository there:

   ```
   apt-mark hold 'openattic-.*'
   apt-get install mercurial
   cd /srv
   hg clone https://bitbucket.org/openattic/openattic
   ```

5. In the file `/etc/default/openattic`, change the `OADIR` variable to point to the clone:

```
OADIR="/srv/openattic"
```

6. In the file `/etc/apache2/conf.d/openattic`, change the `WSGIScriptAlias` line to point to the clone:

```
WSGIScriptAlias         /openattic              /srv/openattic/openattic.wsgi
```

7. Run `oaconfig install`.

You can now start coding in `/srv/openattic`. The openATTIC daemons, GUI and the `oaconfig` tool will automatically adapt to the new directory and use the code located therein.

Mercurial already offers you a full-fledged source control, where you can commit and manage your source code. Please refer to Hg Init: a Mercurial tutorial if you are not yet familiar with this tool.

In order to submit changes to the openATTIC team, refer to *Submitting code to openATTIC*.

8.2 RPC API

The RPC API is the key component that allows the outside world to talk to openATTIC. As outlined in the introduction, its task is to decide which host is responsible for handling a request, forwarding it to the core on that host, and encoding the response in a way that it can be sent back to the caller.

The RPC API consists of three layers:

1. Transport layer
2. Proxy layer
3. Handler layer

8.2.1 Transport layer

Besides the XMLRPC interface, openATTIC also provides the API via Ext.Direct for easier integration with ExtJS, the framework used for the implementation of the GUI.

The transport layer consists of `rpcd/extdirect.py` and the `runrpcd.py` management command. These files provide a Django view and an HTTP server, and are responsible for API module loading and exposing the lower layers via the respective interfaces.

8.2.2 Proxy layer

The proxy layer, implemented by `rpcd.handlers.ProxyModelHandler`, provides the target host detection facility and handles forwarding requests either to the local Model handlers, or the remote host's XMLRPC server.

8.2.3 Handler layer

The handler layer, implemented by `rpcd.handlers.ModelHandler`, provides the actual functionality by initializing the core and calling its methods.

8.2.4 Extending the API

To extend the API, add a module named `rpcapi` to your Django application and make sure it has a variable named `RPCD_HANDLERS`, which contains a list of handler or proxy classes that make up the interface you wish to expose.

The transport layer will automatically pick up your module and add it to the exposed APIs.

8.3 openATTIC Core

The openATTIC core makes heavy use of the Django framework and is implemented as a Django project, consisting of several apps – one for each supported functionality or backend system.

Each app bundles a set of submodules. Models are used to represent the structure of the objects an app is supposed to be able to manage. The *RPC API* is used for interaction with the models. And lastly, the *System API* can be used in order to run other programs on the system in a controlled way.

8.3.1 Models

Models are used to provide an abstraction for the real-world objects that your app has to cope with. They are responsible for database communication and for keeping an eye on the state of the whole system, being able to access any other piece of information necessary.

Please check out Django at a glance for more information.

8.3.2 Filesystem API

The filesystem API abstracts handling different file systems, translates actions initiated by the model into commands to be executed and calls Systemd accordingly.

8.4 System API

The system API handles execution of commands on the system in a controlled fashion. Note that it is *not* responsible for interpreting the output in any way, all interpretation is the job of the higher layers.

8.4.1 DBus Interface

The system API is a DBus RPC API available on the System bus under the name `org.openattic.systemd`.

Note: A tool that simplifies inspection of DBus interfaces is pdbus.

8.4.2 Accessing the System API

In order to make calls to the System API from the core, you can easily retrieve systemd plugins by their path using the `systemd.helpers.get_dbus_object()` function which, when passed a dbus path, returns the object associated with that path:

```
>>> from systemd import get_dbus_object
>>> volumes = get_dbus_object("/volumes")
>>> volumes
<ProxyObject wrapping <dbus._dbus.SystemBus (system) at 0x7fc9a748df50> :1.1973 /volumes at 0x7fc9a8
```

You can then proceed to calling the methods exported by this plugin on the proxy:

```
>>> volumes.write_fstab()
>>>
```

Systemd transactions

When it comes to doing complex tasks like the creation of a volume, usually a whole bunch of commands needs to be ran in sequence in order to implement the task. For this, Systemd provides Transactions.

The client code can use the `systemd.helpers.Transaction` context guard in conjunction with Python's with statement in order to make use of this functionality. When running inside a transaction, certain systemd calls that are marked as deferrable will be deferred until the end of the transaction block. If the transaction block runs into an exception, all the queued commands will be discarded and no action will be taken at all. Otherwise, the queued commands will be executed in sequence. If one command in the sequence fails, the following commands will be discarded as well.

Mocking DBus interfaces for Unit tests

If you adhere to the pattern of using `systemd.helpers.get_dbus_object()` for all access to the System API, mocking it can be easily done using the mock python library. See the ZFS unit tests for elaborate examples of how to do so.

8.4.3 Extending the System API

To extend the API, add a module named `systemapi` to your Django application, which contains classes derived from the `systemd.plugins.BasePlugin` class. All classes that inherit from `BasePlugin` will be exposed automatically.

Paths

DBus is an object-oriented RPC mechanism. In order to identify different objects, each object is associated with a path. This association is defined by setting the `dbus_path` class variable when defining a plugin:

```
class SystemD(BasePlugin):
    dbus_path = "/volumes"
```

Implementing Plugins

DBus uses function signature strings to describe the number and type of arguments required by different functions. Only those methods of your Systemd plugin that are associated with a signature string will be exposed by DBus, so you have to make use of the `systemd.plugins.method()` or `systemd.plugins.deferredmethod()` decorators in order for your methods to be available.

The `systemd.plugins.method()` decorator accepts two signatures: `in_signature` describes the function parameters, `out_signature` describes the structure of the return value. Methods decorated with `method` will always be called directly and cannot be part of transactions.

If you need a method to partake in transactions, you need to decorate it using `systemd.plugins.deferredmethod()`. This decorator only accepts an `in_signature` because methods cannot return anything when ran as part of a transaction (the caller would have no way of retrieving the return value).

The invoke function

Since systemd's main job is the execution of commands, it comes with a function called `systemd.procutils.invoke()` that does the heavy lifting.

8.5 Integration Testing

When making changes to openATTIC, you will want to make sure those changes work as intended and do not break anything. For that, you can use Gatling, the openATTIC integration test suite. Gatling uses the openATTIC XML-RPC API and runs a complete test of all features, making sure they work as intended.

8.5.1 Prerequisites

In order to run Gatling, you need to have a development box handy that supports Python, and you require an API URL. How to obtain the API URL is described in *XML-RPC API*.

8.5.2 Setting up Gatling

The Gatling source code is maintained in a Mercurial repository at https://bitbucket.org/openattic/gatling. To use it, clone the repository to your development system using the following command:

```
hg clone https://bitbucket.org/openattic/gatling
```

This will create a directory named *gatling* which contains the complete source code for Gatling.

Before you can run Gatling, you will have to configure the API URL. To do this, create a file named after your host in the *conf* subdirectory of the Gatling tree, e. g. *conf/srvopenattic01.conf*, and put the following lines into it:

```
[options]
connect = <API URL>
```

Then you can run Gatling using:

```
python gatling.py -t srvopenattic01
```

Note: Depending on your system's general performance and the modules you have installed, this can take a long time.

For more information, please refer to the Gatling documentation.

8.6 Submitting code to openATTIC

So you have written some code that you would like to submit to the openATTIC team? Great, we love contributions! We just need to ask you to follow these few steps.

1. Joining our IRC channel helps us to get to know you and your project. This way, we can guide you through your development and preparation phase.
2. Please add unit tests to your modules.
3. Sign up for a BitBucket account, if you have not already done so.
4. Fork the openattic repository into your own account. Please see the tutorial on how to do so.
5. Push the changes you have commited locally to your fork of the repository. You do not have to fork the repository before making any changes, you can just as easily push changes you made already into a newly-created fork.
6. Send us a pull request.

See also:

This documentation is also available as a PDF file.

CHAPTER

NINE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*
- *fulltoc*